;login:

THE MAGAZINE OF USENIX & SAGE
July 2000 • volume 25 • number 4

inside:

USENIX & SAGE
The Advanced Computing Systems Association &
The System Administrators Guild

USENIX Upcoming Events

3rd Large Installation System Administration of Windows NT/2000 Conference (LISA-NT 2000)

Sponsored by USENIX & SAGE

JULY 30 - AUGUST 2, 2000
Madison Renaissance Hotel, Seattle, Washington, USA
http://www.usenix.org/events/lisa-nt2000

4th USENIX Windows Systems Symposium

AUGUST 3-4, 2000
Madison Renaissance Hotel, Seattle, Washington, USA
http://www.usenix.org/events/usenix-win2000

9th USENIX Security Symposium

AUGUST 14-17, 2000
Denver Marriott City Center, Denver, Colorado, USA
http://www.usenix.org/events/sec2000

4th Annual Linux Showcase and Conference

Sponsored by USENIX and Atlanta Linux Showcase, Inc., in cooperation with Linux International

OCTOBER 10-14, 2000
Atlanta, Georgia, USA
http://www.linuxshowcase.org

First Workshop on Industrial Experiences with System Software (WIESS 2000)

Co-sponsored by IEEE TCOS and ACM SIGOPS (pending)

OCTOBER 22, 2000
Paradise Point Resort, San Diego, California, USA
Web site: http://www.usenix.org/events/osdi2000/wiess200

4th Symposium on Operating Systems Design & Implementation (OSDI 2000)

Co-sponsored by IEEE TCOS and ACM SIGOPS

OCTOBER 23-25, 2000
Paradise Point Resort, San Diego, California, USA
http://www.usenix.org/events/osdi2000

14th Systems Administration Conference (LISA 2000)

Sponsored by USENIX & SAGE

DECEMBER 3-8, 2000
New Orleans, Louisiana, USA
http://www.usenix.org/events/lisa2000

6th USENIX Conference on Object-Oriented Technologies and Systems

JANUARY 29 - FEBRUARY 2, 2001
San Antonio, Texas, USA
http://www.usenix.org/events/coots01

3rd USENIX Symposium on Internet Technologies and Systems (USITS '01)

MARCH 26-28, 2001
Cathedral Hill Hotel, San Francisco, California, USA
Web site: http://www.usenix.org/events/usits01
Submissions due: September 18, 2000

Java™ Virtual Machine Research and Technology Symposium

APRIL 23-24, 2001
Monterey, California, USA
Web site: http://www.usenix.org/events/jvm01
Submissions due: November 1, 2000

2001 USENIX Annual Technical Conference

JUNE 25-30, 2001
Boston, Massachusetts, USA
Web site: http://www.usenix.org/events/usenix01
FREENIX Refereed Paper submissions due:
November 27, 2000
General Session Refereed Paper submissions due:
December 1, 2000

For a complete list of future USENIX events, see **http://www.usenix.org/events**

contents

;login: Vol. 25 # 4, July 2000

;login: is the official magazine of the USENIX Association and SAGE.

;login: (ISSN 1044-6397) is published eight times a year by the USENIX Association, 2560 Ninth Street, Suite 215, Berkeley, CA 94710.

$40 of each member's annual dues is for an annual subscription to *;login:*. Subscriptions for nonmembers are $50 per year.

Periodicals postage paid at Berkeley, CA and additional offices.

POSTMASTER: Send address changes to *;login:*, USENIX Association, 2560 Ninth Street, Suite 215, Berkeley, CA 94710.

cover photo: Ohrid, Macedonia; see p. 75

motd

by Rob Kolstad

Dr. Rob Kolstad has long served as editor of *;login:*. He is also head coach of the USENIX-sponsored USA Computing Olympiad.

<kolstad@usenix.org>

Lists

Are you a list maker? I guess I am.

I'm packing this week to depart on a ten-day scuba trip on Sunday. I'll be living on a boat and travelling far enough from port that forgetting an important item (e.g., a mask) would have a dramatic negative impact on my trip.

Having done this for 20 years, I now have a packing list with 118 items on it. I've used this list for the better part of a decade and I know that I can survive the trip if I just do what it says. Of course, I will probably still wait until the last minute to pack. I hate packing.

I also make "procedural lists." I have two lists for running the regional science fair. By scrupulously following the directions (e.g., "order the blue certificates on February 1"), I know that I have a good shot at making the science fair succeed. I tweak this list only slightly on an annual basis.

I use lists as stress-reducers. Once I know they work, I can feel confident that the "details" are taken care of and that I can concentrate on the task at hand instead of multiplexing a big-picture worry ("Do I have enough . . . ?" for example).

You can make lists a number of ways. For my packing lists, I do the best I can and then itemize the things I've packed. As I go through the trip (or event, or whatever), I augment the list. As I hear what other people have on their lists, I judiciously plagiarize their best ideas.

As for procedural lists, they require a bit more discipline, especially for events that happen just once a year (like a science fair). When I am to run such an event, I keep an extra window open with nothing but a vi of the procedural file. Each time I take a step (check a file, call someone, write a letter, etc.), I record it.

When running the event from the list, I allow myself only to execute instructions on the list (which means I edit the list in realtime to add forgotten instructions). About one pass of realtime execution and the list is a guaranteed stress reducer for the future.

I worked with a conference organizer whose lists ran to 150 pages per conference! This makes sense to me, since conferences have a huge number of details that must be tracked.

I found out that some people eschew lists. I lent some equipment to one traveller who failed to return some of it. Why? "It's just not their style," replied a friend. "They don't think that way." What a surprise! I had no idea that people enjoy the challenges that result from trying to solve anticipatable problems in realtime. I gave that up years ago – way too much stress when one is already maxed out!

I like lists, though they do strike me as being somewhat anal-retentive. I'll continue to use them as stress reducers. I just hope I'm not pushing myself too much toward being one of the automatons that Bill Joy fears so much in his *Wired* article.

apropos

"Hot Spares" For DoS Attacks

by Tina Darmohray

Tina Darmohray, co-editor of ;login:, is a computer security and networking consultant. She was a founding member of SAGE.

<tmd@usenix.org>

and Ross Oliver

<reo@iwi.com>

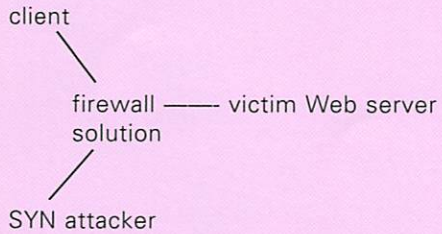
Recent denial-of-service (DoS) attacks have proven to be challenging for even the most well-secured sites. Using distributed commandeered systems to launch the attacks has resulted in a "win-win" scenario for the attackers: they benefit from access to more horsepower than they own directly, and they cover their tracks more thoroughly, since a trace back to the attacking machines simply leads to the other set of victims, the commandeered machines. With this kind of leg-up to contend with, what kind of countermeasure can organizations use to defend against this type of attack?

As with most computer-security problems, there is no single silver bullet that will defend against a DoS attack, but there are ways you can mitigate the problems. The CERT Coordination Center Denial of Service Tech Tip (*<http://www.cert.org/tech_tips/denial_of_service.html>*) suggests that sites invest in "hot spares" that can be pressed into service if a similar machine is disabled by an attack. Another "hot spare" approach is to have an arsenal of SYN protectors that can be deployed rapidly if needed. Ideally, these are firewalls that can sit in front of your Internet resources and take the brunt of the SYN flood attacks.

We're researching a number of firewall solutions to determine what kind of protection they provide against SYN flood attacks. We determined that some of them offer value-added features that deflect SYN floods. One way they do this is by proxying the requests and not passing them on to the victim server unless the 3-way handshake is complete.

In order to test the firewalls we created a test environment to simulate the DoS attacks we'd been seeing in the last several months. Our environment consists of a Linux 2.2.12-20–based attack machine, an NT client machine to fetch Web pages, a Linux 2.2.12-20–based victim Web server to serve up Web pages, and a firewall solution in the middle. Our SYN flooder is a proprietary program that is designed to achieve the maximum possible SYNs/second. Using the shortest Ethernet frame size of 64 bytes, the maximum rate of a 10Mb Ethernet is 14,880 pps. Our SYN packet is padded to the minimum Ethernet (RFC894) packet size, so the maximum SYN flood rate of a 10Mb Ethernet is ~14,880 SYNs per second, and a 100Mb Ethernet is ~148,800 SYNs/sec.

Test Environment:



We created a script on the client system to request a set of Web pages from the victim Web server and measured the elapsed time to get the pages under normal network conditions. Then, to test the effectiveness of the firewall solution, we launched the SYN flood attack and determined what effect it had on the elapsed time to get the pages. We used a Checkpoint to test a traditional firewall passing the SYNs on to the victim Web server, and a NetScreen-100 to test a solution which proxies SYN requests. Here's what we observed:

DoS demo against traditional firewall solution [Nokia running Checkpoint]:

Output from the client attempting to fetch Web pages from the victim Web server:

```
C:\DEMO>fetch web-pages
Fetching 21 HTML pages, total 35K of data...
Elapsed time: 2.78 seconds
Fetching 21 HTML pages, total 35K of data...
Elapsed time: 2.61 seconds
Fetching 21 HTML pages, total 35K of data...
Elapsed time: 2.81 seconds
Fetching 21 HTML pages, total 35K of data...
Elapsed time: 2.18 seconds
```

[The SYN flood attack was launched here.]

```
Fetching 21 HTML pages, total 35K of data...
Elapsed time: 26.02 seconds
Fetching 21 HTML pages, total 35K of data...
Elapsed time: 36.07 seconds
```

[The SYN flood attack ended here.]

```
Fetching 21 HTML pages, total 35K of data...
Elapsed time: 2.68 seconds
Fetching 21 HTML pages, total 35K of data...
Elapsed time: 2.39 seconds
Fetching 21 HTML pages, total 35K of data...
Elapsed time: 2.70 seconds
Fetching 21 HTML pages, total 35K of data...
Elapsed time: 2.79 seconds
Fetching 21 HTML pages, total 35K of data...
...
```

Output from the attack machine showing SYN flood rate (each "%" represents ~500 SYNs and is displayed in realtime on the attack machine):

```
% flood firewall
Rate: 495 SYN/sec
%%%%%
Rate: 495 SYN/sec
%%%%%
Rate: 495 SYN/sec
%%%%%
Rate: 496 SYN/sec
%%%%%
Rate: 495 SYN/sec
%%%%%
Rate: 496 SYN/sec
...
```

DoS demo against proxied-SYN solution (NetScreen-100):

Output from the attack machine showing SYN flood rate:

```
C:\DEMO>fetch web-pages
Elapsed time: 1.29 seconds
Fetching 21 HTML pages, total 35K of data...
Elapsed time: 0.99 seconds
Fetching 21 HTML pages, total 35K of data...
Elapsed time: 0.93 seconds
```

Recall that SYN flood attacks succeed by exploiting the TCP three-way handshake. To establish a TCP connection, two machines negotiate a triple exchange consisting of:

1. the connection initiator sending out a SYN packet
2. the receiver acknowledging with a SYN/ACK packet
3. the initiator completing the handshake by acknowledging the receiver's reply with her own ACK.

By flooding a machine with illegitimate initial SYN packets that won't be acknowledged, you can exceed a system's limit of connections that are waiting to be established. In this way you can prevent a machine from being able to respond to any legitimate connection requests.

```
Fetching 21 HTML pages, total 35K of data...
Elapsed time: 0.95 seconds
Fetching 21 HTML pages, total 35K of data...
Elapsed time: 0.92 seconds
Fetching 21 HTML pages, total 35K of data...
```

[The SYN FLOOD attack was launched here.]

```
Elapsed time: 0.96 seconds
Fetching 21 HTML pages, total 35K of data...
Elapsed time: 0.94 seconds
Fetching 21 HTML pages, total 35K of data...
Elapsed time: 0.94 seconds
Fetching 21 HTML pages, total 35K of data...
Elapsed time: 0.94 seconds
Fetching 21 HTML pages, total 35K of data...
Elapsed time: 0.94 seconds
Fetching 21 HTML pages, total 35K of data...
Elapsed time: 0.91 seconds
Fetching 21 HTML pages, total 35K of data...
...
```

Output from the attack machine showing SYN flood rate:

```
% flood netscreen
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Rate: 13869 SYN/sec
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

Rate: 13927 SYN/sec
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
...
```

With the traditional firewall in place, the victim Web server held up to about 500 SYNs/second. Any higher rate of SYN flood rendered the victim Web server completely unable to answer requests. When protected by the NetScreen-100, the victim Web server showed no noticeable change in ability to serve Web pages at ~500 SYNs/second. We continued to turn up the SYN flood rate to determine when a Web server protected by the NetScreen-100 would begin to be affected by the SYN flood. Our tests show that the victim Web server begins to degrade when the attack goes above ~14,000 SYNs/second.

We concluded that proxying connection requests offers a significant improvement in protection against SYN flood attacks. We plan to test additional firewalls to determine if they are doing something interesting with SYN handling and if one approach is better than the rest. Check *<http://www.iwi.com>* for test result updates.

;login:

EDITORIAL STAFF

EDITORS:
Tina Darmohray <tmd@usenix.org>
Rob Kolstad <kolstad@usenix.org>

CONTRIBUTING EDITOR:
Rik Farrow <rik@usenix.org>

STANDARDS REPORT EDITOR:
David Blackwood <dave@usenix.org>

MANAGING EDITOR:
Jane-Ellen Long <jel@usenix.org>

COPY EDITOR:
Eileen Cohen

TYPESETTER:
Festina Lente

MEMBERSHIP AND PUBLICATIONS
USENIX Association
2560 Ninth Street, Suite 215
Berkeley, CA 94710
Phone: 510 528 8649
FAX: 510 548 5738
Email: <office@usenix.org>
WWW: <http://www.usenix.org>

letters to the editor

FOR THE LOVE OF THE GAME

FROM STEVE SHAW
<sshah@planetoid.org>

Hi Tina,

I read your article in the April, 2000 *;login:* and I must agree – putting in 60 hours a week has more to do with loving a good challenge and less to do with money. (Although the money sure helps! Esp. living in Silly-con Valley!) I've been reading your articles for a while now and I must say that it is nice to see less ego and more common sense coming from UNIX admins. (My wife, also a member of SAGE, really appreciates that component.)

ANOTHER REMINISCENCE

FROM PETER SALUS

My notes and those of Ted Dolotta spurred a respectable amount of reminiscence on the part of colleagues. This note from Aharon [Arnold] Robbins in Israel adds to the historic narrative.

I read the stuff in the April 2000 *;login:* about System IV. I can perhaps add a bit more info for you. Feel free to publish this.

In the summer of 1982, while in grad school, I did some contract programming at Southern Bell. At the time, Southern Bell was still part of the "Bell System." Although the public didn't see it, there was lots of internal activity and anticipation towards "1/1/83," when divestiture was set to happen.

I did my work on a PDP-11/70 running UNIX 4.0. The doc I got was the System III documentation; the 6 x 9 reference manual and the two-volume 8.5 x 11 set of papers. I was able to keep the reference manual, and I still have it in a box somewhere. During my lunch breaks, I would just sit and read the manual. I never learned so much in so short a period of time.

(As a side note, the Research V8 manual was printed in 6x9 format. For V9 and V10 [and Plan 9], they went back to 8.5 x 11.)

The following are from memory. It seems that for 4.0 they didn't reprint the whole documentation set, they just sent out release notes that described what had changed. Most notable was that a number of internal kernel tables were changed to use hashing instead of linear search, thus speeding up the system, e.g., the inode table.

A humorous item I remember was that "the destroy your input file feature of sort(1) has been removed." It seems that in System III, sort's -o option to send the final output to one of the input files would open and truncate the file before reading the input. Ooops.

Another notable feature of the System III/IV doc was that it described UNIX as "an operating system for the Digital PDP-11 and VAX systems and for the IBM Series/370 architecture computers" (paraphrased).

And indeed, if I remember correctly, there was a subdirectory in the kernel source for the 370. (An article in the October 1984 Bell Labs Technical Journal, if I remember correctly, discusses what was done to put UNIX on the 370.)

Finally, the group at Southern Bell was using an experimental screen editor named se. To my knowledge, it was never released outside the Bell System. To squeeze se onto the PDP-11, they took out support for regular-expression searching. I decided not to bother with it, and wrote my several thousand lines of code in ed.

In any case, at the time, the release policy from Bell Labs was that external releases were one revision back from whatever was current inside the Bell System. Thus, System III was released to the world, but internally, 4.0 was in use.

more letters to the editor

I remember when the group I was working for got the letter discussing the upcoming 5.0 release. It included a statement that with that release, the Labs would be changing their release policy, and shipping the same version to both internal and external customers. This was undoubtedly motivated by the fact that AT&T would soon be allowed to be in the computer business for real, but I don't remember if the 5.0 announcement letter explicitly made that connection or not.

Thus, if not for the policy change, 5.0 would have gone to the Bell System sites, and 4.0 would have been released to the world. But since they wanted to have everyone be current, 4.0 was never released outside the Bell System.

LETTERS TO RIK FARROW

FROM SERGEY BABKIN
<sergey@sco.com>

In your "Musings" article in the April 2000 *;login:* you mentioned contacting the systems administrators internationally. I had no experience of international attacks but I've been a sysadmin in Russia (exactly one of your hypothetical cases). So I guess I can provide some information on the issue.

From my experience virtually all sysadmins in Russia have knowledge of written English to some degree. The reason is that much of the software documentation and software itself is in English, so some knowledge of written English is essentially a professional requirement. I suppose the situation may be very much the same in the rest of the world. So sending email to a foreign sysadmin should not impose big problems. Just use simpler grammar and vocabulary to reduce the chance that your message would be misunderstood. Of course a chance that this particular person does not know English at all does exist but it's quite thin.

The situation with contacts by telephone is quite different. First, the time-zone difference. The time-zone difference between the U.S. and Russia would be somewhere between 7 and 15 hours, so there is a big chance that your call will be at 2:00 a.m. over there and nobody will be in the office to answer your call. Second, the average knowledge of spoken English among the sysadmins is rather close to zero. English is known for its loose connection between writing and pronunciation. So speak slowly and spell out the words which cause problems with understanding on the other side. British accents are commonly much more difficult to understand to the Russians (and I suppose many other foreigners too) than the American accent. Also the accent may be unexpected for both sides – even if the sysadmin on the other side is able to understand what you're saying you may need to apply some effort to understand the reply (matching what you hear to the spelling instead of traditional English pronunciation may help a lot in this process). Third, the contact telephone given in the WHOIS database may be the general telephone of the IS department or a common telephone shared per room. So an additional step of passing the call to the sysadmin would be added. And there is not really a high chance that the person who answered the phone can talk English too. So my general recommendation would be: don't use the telephone; send email instead.

One problem is that under massive denial-of-service attack the email may not be working at all. Luckily for the victims of denial-of-service attacks, the throughput of international connections is rather limited (and that's especially true for poorer countries), so mounting a denial-of-service attack directly from hosts over there is problematic. And in many cases they would congest the international connections earlier than the victim's connection, so the network admins of the local providers would independently notice the problem (maybe also earlier than the targeted victim) and start working on it. And of course the telephone has the advantage of realtime conversation. So it may be very difficult to replace it with email.

One way to make at least some minimal authentication using the telephone is to ask for the number to call back. But with international calls things become complicated: in many cases it's unreasonable to expect that the person in Russia would call back to the U.S. The reason is that the long-distance rates are much higher there (calls to Russia cost $0.28/min, calls from there are $2–4/min depending on time of day and that's not that far from a typical daily wage), organizations have tighter budgets, and often all the long-distance calls have to be explained to the management. So getting permission for an international call may require going through a few levels of managers who most probably won't be enthusiastic, and in academic organizations that may be not possible at all due to limited budgets and high level of bureaucracy.

A little addition to the list of the telephone troubles: if the telephone number is used by many people, then getting the right one may include the difficulty of pronouncing his/her name right. Spelling of names between the languages is complicated by the fact that different languages use different sets of sounds, so some approximations must be used, plus the special peculiarities of English spelling. So reading a name in a way understandable by an arbitrary person speaking only the native language may be far not as easy as it seems at first sight.

Hope this information could help someone.

RIK REPLIES:
In realtime incident handling, it is better to use the telephone than to attempt to use email. If a site has been compro-

letters . . .

mised, so might the email (or perhaps it will be monitored). A compromise solution would be to attempt to make contact via phone, and continue the conversation using email (unless the network is unusable due to DoS). This actually strikes me as a very good idea, as the person attempting a live trace-back might be faster at scanning netstat-a output sent with the email than the person reached via the phone.

Using the phone instead of email has its own set of problems. Social engineering is based on nontechnical means of gathering information, for example, via telephone calls. Your suggestion, having someone call back, is often used as a means for identification, although it can be spoofed as well. Having worked with some people who have attempted (and sometimes succeeded) in live trace-backs, even getting people to cooperate at all can be difficult. I found that the people who had the most success worked for the armed forces, and commercial sites had less success (keep in mind that this is anecdotal information).

FROM SCOTT DORSEY
<*kludge@grissom.larc.nasa.gov*>
I liked your article in the April 2000 *;login:* quite a bit, but I think you missed an important part about SYN attacks, that they basically depend on a design problem with the Internet protocol. The one clear and obvious solution to specifically prevent SYN attacks isn't RFC2267 filtering or legislative action, but by fixing the problem where it starts. And that means IPv6, I am sorry to note.

RIK REPLIES:
While the authentication header (AH) in IPv6 could help, it relies on the existence of ubiquitous PKI, which seems to be farfetched at this moment in time. Also, most DDoS tools write directly to raw sockets, so they can add whatever they want to an authentication header. Just checking the AH would take a lot of time, as it involves looking up the public key of the alleged host, and checking the digital signature.

In a separate conversation with Eliot Lear, he reminded me that the IETF has been looking into this area, and Steve Bellovin has suggested a mechanism called ITRACE, adding one packet with every 20,000 in a new ICMP type containing forwarding information. There are problems with this too (the additional overhead for routers).

In the latest dissection of a denial of service tool, stream2 (see Dave Dittrich et al. <*http://staff.washington.edu/dittrich/misc/ddos/*> for these analyses), the authors point out that while RFC 2267 may spare remote targets, the internal network may still become unusable. Of course, if you performed 2267-style filtering throughout your network, as the University of Minnesota does, you would avoid this problem as well.

FROM PETER FASSBERG
<*pf@leissner.se*>
I've read your "Musings" article in the April 2000 *;login:*. You asked for people who have experienced some situations.

In August 1998 I tracked down a hacker who had broken in to one of our customers' systems. About one week after that my mailbox was swamped every day. . . . I got about 200,000 "User unknown, returned to sender" mail, but also about 200 mails from angry sysadmins – a couple of them said they should kill me!

That damned hacker had sent SPAM (offering SPAM CDs) to some 2 million email addresses. With MY ADDRESS as From: and Reply-to: !!!

He then used about 25 SMTP servers as relays, all in different parts of the world. . . . None in U.S. or Sweden . . . Tokyo, Moscow, Paris, Gibraltar . . .

Guess how easy it was to tell a sysadmin at a nuclear power plant in Japan that their system was used as a spam-relay ? Puhhh . . .

The sad thing was that 20 of those 25 didn't answer at all!

It lasted for about four weeks. He was shut down by two or three ISPs in the U.S. and I haven't heard of him since then.

I sent about 200 people mail with instructions on how to read email headers. Everyone blaming me for sending those mails could have looked a little at the headers. . . . Here are the URLs I sent to all complainers:
<*http://spam.abuse.net/*>,
<*http://spam.abuse.net/howtocomplain.html*>.

I used ELM at that time, and it crashed my mailbox eight times in that period.

FROM BEN ELLISTON
<*bje@air.net.au*>
I read your article in a recent issue of *;login:* about international break-ins and you asked to hear from anyone who has had to deal with one.

While my break-in late last year was not from Russia or Indonesia, it was still equally challenging. My system (in Canberra, Australia) was broken into by a user connected to an Internet service provider in Mountain View, California.

I won't go into the details, but suffice it to say, I set up a packet sniffer and was able to predict them logging in a second time. I got their IP address and, assuming it wasn't spoofed, used WHOIS to discover the network contact. I called them on the phone and they were quite surprised to hear from someone in Australia! It worked, though — I don't think things would have been handled quite so well if I had used email to contact the ISP. In this day and age, it seems like making a telephone call is seen as being a somewhat drastic measure.

more . . .

HACKERS AND CRACKERS

FROM JOSEPH E. WALS
<crli@crli.com>

I recently became a member of both USENIX and SAGE, and have received my first issue of *;login:* (April 2000). I must say that I'm quite impressed with the magazine. I've joined many professional organizations over the years, and few have had such a truly useful periodical. I found the whole issue useful, but in particular, I enjoyed Oleg Kiselyov's article "Speaking HTTP" for its insight into that protocol's possibilities, Clif Flynt's "The Tclsh Spot" for its really useful hack, Joseph N. Hall's "Effective Perl Programming" for its insights into CGI security issues, and Peter H. Salus's "20 Years Ago in UNIX" for his informative yet laugh-out-loud look at history.

There is, however, one area of improvement that I would like to see addressed in future issues. Please use the term "hacker" in its proper way. I'm sure you folks have been around long enough to know that a hacker is someone who enjoys exploring programmable systems, while someone who enjoys ruining programmable systems is a "cracker." I found the misuse of the terms particularly egregious in Marcus J. Ranum's article, "The Network Police Blotter." Sadly, I've grown to expect the ignorant mass-media outlets to use the wrong term, but I expect better of trusted and informative computer journals such as yours.

Thank you for your time.

TINA REPLIES:

I noted a similar discussion about the use of these terms in a sidebar in the very popular new book *Hacking Exposed.* They seemed to imply that the media has all but obliterated the chances of picking at this nit successfully. They may be right when even the "father of the firewall" Marcus Ranum isn't making any differentiation between the two.

THANKS!

FROM YURAN LU
<ylu71@maine.maine.edu>

I attended the Balkan Olympiad in Informatics in Macedonia this May, as a member of the U.S. team. Our team had a great time, and a wonderful learning experience. It was my first trip abroad for a competition, and I met many new friends internationally, and have been fortunate to hear about how they have grown up to learn programming, as well as other subjects.

Culturally enriching experiences like these give us all incentive to work hard to perform well. I would like to thank [USENIX] greatly for sponsoring this wonderful trip for the U.S. team.

[See the results of that competition on page 75]

Conference Reports

This issue's report is on the USENIX-sponsored 10th Conference on Computers, Freedom, & Privacy, held in Toronto, Ontario, April 4–7, 2000.

Thanks to Lorrie Cranor (<*lorrie@research.att.com*>) for coordinating these reports, and to the summarizers: Anne Adams, Brett Burney, Kate Crabtree, Kat Hanna, Mark Kerr, Mathias Klang, Megan McCormick, Alexander Macgillivray, Lauren Matheson, Ernest Miller, Mark Hissink Muller, Thomas Nauer, Nadia Olivero, Kayvan Sadeghi, Kurt M. Saunders, Lina Tilman, David Todd, Marc Waldman, Alma Whitten, and Sarah Wilford.

The 10th Conference on Computers, Freedom, & Privacy
April 4-7, 2000
Toronto, Ontario, Canada

WORKSHOP ON FREEDOM AND PRIVACY BY DESIGN

Summarized by Marc Waldman

Can a system be designed in such a way that it guarantees strong protection of civil liberties? That was the main theme of the Workshop on Freedom and Privacy by Design, held on April 4th.

The workshop, chaired by Lenny Foner, brought cryptographers, programmers, and system architects together with experts on the issues of freedom and privacy for the purpose of discussing how to design systems that can secure civil liberties. The workshop discussions revolved around three projects. The first project was a proposed replacement of the domain name system (DNS). DNS is a hierarchical naming scheme that is primarily used to translate domain names into IP addresses used to route packets of data around the Internet. Although DNS works very well, it has certain characteristics that make it less than ideal from a civil-liberties point of view. The unique nature of domain names allows the first company or individual that purchases a domain name to have complete control over it. This leads to "land grabs" whereby a company purchases all domain names that are in some way connected to its business. This results in rapid exhaustion of the DNS namespace and also prevents smaller companies or individuals from acquiring these domain names. The unique mapping of domain names to IP addresses also makes anonymous publishing very difficult. The DNS system does not provide a mechanism to hide the true IP address of a particular domain name.

The DNS-replacement discussion consumed the morning session. Rebecca Wright of AT&T Research Labs described the challenges and obstacles to building such a system, including complying with existing laws and industry standards. A particular system, in order to gain widespread acceptance, must be easy to use.

Alma Whitten of Carnegie-Mellon University gave a brief talk concerning user-interface design for privacy-enhancing technologies. Whitten stated that developers of privacy-enhancing technologies all too often expect the average user to understand complex topics such as key distribution and digital signatures. An incomplete understanding of these topics can lead an individual unknowingly to reveal sensitive information like secret passwords.

After the two brief talks was a moderated discussion of the DNS system. Discussion topics included the need for a hierarchical naming system, the role of search engines, and the need to preserve the underlying DNS.

The second project concerned ways to motivate businesses to protect their customers' civil liberties. David Phillips of the University of Texas at Austin discussed the activities of anti-nuclear activists and how they could be adapted by civil-liberties activists. John Gilmore of the Electronic Frontier Foundation discussed the free software movement and described starting the free-software company Cygnus.

The moderated discussion that followed focused mainly on ways of convincing businesses to protect individual privacy better. There was also discussion of what form a privacy "Chernobyl" would take – that is, an event that would cause average users to demand greater privacy protection from the companies they deal with.

The third project concerned "anonymous" cash – cash that cannot be traced back to a particular individual. Companies that issue credit and debit cards routinely use data-mining tech-

niques to discover the buying habits of individuals. This allows them to target advertising at the individual who owns the card. Anonymous cash systems would prevent this type of targeted advertising, and therefore the credit-card companies have little incentive to get involved with anonymous cash. The moderated discussion suggested various methods to coax credit-card companies to issue anonymous cash cards.

PANEL: DOMAIN NAMES UNDER ICANN: TECHNICAL MANAGEMENT OR POLICY CHOKEPOINT

Summarized by Kate Crabtree

The Internet Corporation for Assigned Names and Numbers has had an active year and a half since its formation in 1998. In its mandate, it was formed to "coordinate the management of only those specific technical, managerial and policy development tasks that require central coordination" – the assignment of globally unique names, addresses, and protocol parameters. It was structured to allow for participation in its limited mandate by all stakeholders.

ICANN can lead to spirited debate on issues regarding the privacy implications of the "Whois" Database; the free-expression and intellectual-property implications of the handling of "famous names"; new Generic Top Level Domains (gTLDs); and participation in the ICANN process. On one side of the process issue are those who feel ICANN's role should be narrowed further to include only technical management of domain names and on the other are those who feel that ICANN deserves the opportunity to prove that it is a workable solution for domain-name management.

"It's better than most of us think" was the concluding statement by moderator Michael Froomkin, referring to ICANN, during this panel, before he passed the discussion on ICANN's management of the domain name system on to Jerry Berman of the Center for Democracy in Technology (CDT). Berman would concur with that statement, provided that ICANN heed CDT's recommendation that a more direct and democratic electoral process be followed in electing its at-large board members. He felt that ICANN "is here to stay."

Panelist Karl Auerbach insisted that the image of ICANN as a corporation formed by the Internet community "is a fiction," fundamentally flawed, since its growth thus far has not yielded wide participation by the Internet community. Richard Sexton added his suspicion that ICANN would fail just as many other previous Internet initiatives have. ICANN Board Member Amadeu Abril i Abril, teleconferencing into CFP2000 from Barcelona, expressed confidence that ICANN is a workable and global solution to domain-name management. The internationalization of ICANN is one of his priorities as a board member.

There appeared to be some consensus among the panelists that the U.S. government's role in ICANN should eventually desist in order to allow the global Internet community to appreciate its potential. One reason many of the panelists agree to work within the parameters of ICANN in its current form toward a solution amenable to the Internet community is that if ICANN does not succeed, its failure could lead to the domain name system being managed by governments or the United Nations – not a desirable solution for the Internet community.

PANEL: NEW JUSTICE INFORMATION TECHNOLOGIES: DOES EXISTING PRIVACY LAW CONTEMPLATE THEIR CAPABILITIES?

Summarized by Mark Kerr

What happens when you put together a panel about Big Brother, give seats to suspected representatives of The Man, throw in a couple of civil rights advocates, and then locate the discussion at a privacy-and-freedom conference? You end up with a frank discussion about where government technology is going and how this may affect privacy.

One of the many things the panel was able to do was cut through some of the hype. As Neal Stephenson would recommend later in the evening, we would be given more information to reevaluate Big Brother's place in our threat model. For example, privacy advocates are fond of pointing out that out of the hundreds of wiretaps requested by the FBI in the last decade, none has been rejected by a judge. It is about time that we have had an FBI agent present to point out that three applications (yes, three) were finally turned down last year.

More important, Paul George was able to point out that the reason for the low rejection rate has less to do with gullible judges and more to do with the very strict internal guidelines and review process that a U.S. attorney must go through before a wiretap application will even be approved by the Department of Justice. Even so, Jim Dempsey made a strong case that wiretaps are an area to watch: the FBI system of information gathering is growing, and the potential to gather and analyze information will increase substantially. Wiretaps, having increased 12% in 1998, could increase by 300% with the implementation of Digital Storm, a new FBI program to bring digital technology to bear on wiretap problems.

Beyond such fun facts, the panel showed that future technological choices will have to balance privacy/security with efficiency/convenience. One such balancing act comes with the push for integration of government databases. Judge Tom Cecil, while wanting greater integration within the justice system, recognized that such compilations do increase the risks to citizens' privacy. Integration

cuts through past systematic inefficiencies built into trying to obtain information from the government. Instead of having to go to the proper room in the proper building to fill out the proper paper work, the fear is that it will soon be possible to access from anywhere such diverse public records as a person's divorce proceedings and his car registration.

While ease of access is seen as an inherent threat to privacy, the panel identified specific concerns: when access is combined with an enhanced, integrated government database, the potential for data mining to perform either marketing or fraud grows substantially. Private abuse is only the beginning of what the panel covered; intergovernment abuse of privacy may grow. A practical example of this is the use of biometrically encoded ID cards to reduce welfare fraud. If the creation of such a card system entailed the implementation of a fingerprint database, criminal investigators could infringe on privacy rights by using the welfare databases in investigating crimes. Because of the potential of such abuses, George Tamko argued that it is not enough to pass laws to prevent unauthorized secondary uses of biometrics; the system must be built to restrict access by leaving the biometric information with the user. This can be done by encoding the biometric on the card (or trusted hardware device) and then using it to encrypt PIN numbers. Having the user retain her biometric information will allow for identification without the creation of a centralized database that would be open to abuse.

Audience questions showed that as the government improves its technologies, it too, will be watched.

HOW TECHNOLOGY CAN ENSLAVE US

Steve Talbott, Author and Editor

Summarized by Kayvan Sadeghi

Luncheon speaker Stephen Talbott spoke to a technology-friendly crowd Wednesday. He asked the group to step back from the questions of what can be done and how to do it, and invited us to consider technology in the context of a complex society in which it is likely to have both positive and negative consequences. After citing numerous examples of our "sleepwalking with technology," he proposed a new arena for the privacy discussion. "It's not about the government. It's not about corporations. It's about whether or not we can find it within ourselves to be free." He pointed to the many pessimistic predictions of our future and argued against this fatalistic approach, saying that the course of events will be of our choosing, and he implored us to choose wisely.

Talbott laid the blame for many of the problems he mentioned on a focus on technology as an end in itself, and not a means to more fundamental human goals. "Technology can matter tremendously, but only when you're talking about something else." Talbott did not propose a halt to technological progress, but instead suggested actively engaging our technology and asking "why?" before racing to produce and adopt the newest technology.

Once the floor was open to questions, it became clear that much of the audience was still skeptical, and others wondered how to apply his advice. By attending this conference, however, the audience demonstrated that they already shared his concerns. We are all concerned about the impending (and existing) negative consequences of technology for freedom and privacy. We have gathered precisely to engage technology actively in light of its social ramifications, in hopes of preventing "sleepwalkers" from walking toward a future in which the human ideals of freedom and privacy fall victim to ever-increasing technological capability.

INVITED TALK

Mozelle Thompson, FTC Commissioner

Summarized by Lauren Matheson

"It is an exciting time to be at the FTC," stated Mozelle Thompson after his speech, when he sat and spoke with interested listeners for well over an hour and a half before going to dinner. Thompson's speech and his after-speech remarks focused mainly on the issue of privacy, since the FTC is currently the closest thing the United States has to an established privacy commission.

It was well noted that the U.S. was absent from the panel of privacy commissioners who spoke right before the FTC commissioner. Thompson himself recognized that the U.S. lacks a formal organization devoted to the issue of individual privacy, but stated that the FTC has been active in the relevant areas of concern. He noted that the U.S. does not actually have direct legislation concerning individual privacy, as other countries do, but that existing case law has repeatedly reiterated our presumption of privacy in the U.S.

Several times Thompson passed off the question of whether legislation is "the answer." More than a few people tried to push Thompson into a corner on the legislation issue, asking whether it was going to happen now or later. Thompson refused to answer, because (1) the issue is fairly controversial, and (2) the Internet is still in its infancy and we need more input – such as the expected report from the recently formed Online Advisory Committee. Thompson said he expects to see the report sometime around June.

Instead of dwelling on the legislation question, Thompson focused on education. He declared that we need education

both for businesses and for consumers, i.e., individuals surfing the Net. We should concentrate on making consumers aware of how information about them is being used and gathered when they surf particular sites. But we also need education on the business/corporate side – Thompson suggested that businesses could take an example from IBM and Microsoft, which do not advertise on Web sites that do not have a stated and enforced privacy policy for their surfers.

After discussing the above "policy" issues, Thompson then switched to the FTC's primary function of enforcement – citing cases such as GeoCities and the pending issues with Yahoo! and DoubleClick. He pointed out that problems like these cannot be solved by the FTC alone, but that all levels of enforcement must interact together to create a successful level of privacy for the Internet society. There must be interaction among individuals, businesses, and of course the government. And it doesn't stop there. For enforcement to be effective in a wired world, there must be cooperation among different countries. Thompson at one point stated that "privacy is a very important issue, but it cannot be viewed in isolation."

Thompson stayed around the foyer for quite a while and was kind enough to answer everybody's questions and exchanged business cards with anyone who offered. This is when he made the statement that working at the FTC is a pretty exciting business these days. He noted that what is happening in the online-privacy realm today is the result, not of laws, but of companies working together, mostly in their own interests. It may not be the best situation, but Thompson reserved comment on what could or would happen in the near future. He stated that fraud is fraud, no matter where it happens – on TV, over the telephone, or on the Internet – and that the FTC is committed to protecting against it.

Before rushing off to dinner, Thompson made an important statement to the last remaining listeners about the general purpose of conferences such as these: "It's not about the technologies, it's about the policies." Thompson explained that the technologies will take care of themselves. It's the policies that need to be established and worked out before our online rights get lost in the techno-shuffle.

DINNER SPEECH

Neal Stephenson

Summarized by Marc Waldman

During his talk, science-fiction author Neal Stephenson pointed out that it is now possible to monitor your home from just about anywhere in the world via the Internet. Of course one needs a persistent connection to the Internet, Webcams, and perhaps other Internet-enabled devices, but these are all minor details. The main concern is privacy. While it is great to be able to monitor your own home over the Internet, you want to prevent others from monitoring your home as well. Secret-key mechanisms can be used, but Stephenson pointed out another alternative. Instead of simply storing the Webcam images on your own PC, why not store them on several Web servers, perhaps one owned by law enforcement or a security service. This brings us back to the privacy problem. While we want law enforcement to be able to view a picture of someone committing an illegal act, we may not want law enforcement to view all our Webcam pictures.

Stephenson's solution is to use a technique called secret sharing to store so-called shadows on each of the Web servers rather than storing the Webcam image itself. Secret sharing is a technique that is used to split a password (or any collection of bits) into n pieces called shadows or shares, such that only k of them are necessary to reconstruct the original item – the password in this case. The value k can be less than or equal to the value of n. For example, a password can be split up into ten shares such that only five of them are needed to reconstruct the password. Combining less than k shares does not reveal the password. Instead of the password, Stephenson suggests we secret-split the Webcam image itself. In this scenario, only one share is stored on the law-enforcement Web server instead of the whole Webcam image. This can safely be done, because the solitary share is useless by itself. However, if we want law enforcement to view a particular image, we just send them the other $k - 1$ shares.

Although Stephenson suggested this only as a possible project, a system named Publius already does something similar and incorporates several unique WWW publishing mechanisms. Publius is a censorship-resistant, tamper-evident, WWW-based publishing system. It was designed by Lorrie Cranor (AT&T Research), Avi Rubin (AT&T Research), and Marc Waldman (NYU computer science department). Publius allows an individual to publish static WWW-based content (HTML, PDF, GIF, JPG, etc.) on several servers at once, such that each server cannot tell the type of content it is hosting and any modification to the stored content can be detected. Publius utilizes a secret-sharing mechanism but not in the way described by Stephenson; however, the effect is the same. Publius is written in Perl and will soon be freely available for download at *<http://www.cs.nyu.edu/~waldman/publius.html>*. A paper describing Publius is also available at the same URL. Please send any questions concerning Publius to Marc Waldman (*<waldman@cs.nyu.edu>*).

KEYNOTE ADDRESS: GLOBAL SURVEILLANCE/THE EVIDENCE FOR ECHELON

Duncan Campbell, Investigative Journalist and TV Producer

Summarized by Anne Adams

The use of surveillance to curtail our freedom by controlling and manipulating socially unacceptable behavior is not a modern-day invention. Jeremy Bentham (1832) argued for control by surveillance in the Panopticon, a prison design wherein all the people incarcerated may be watched from a central tower. Although people were not to be watched all the time, they would maintain acceptable standards of behavior for fear of being watched. Fear would be maintained by examples being made of odd individuals to keep the others on their toes. Over the past twenty-five years Duncan Campbell has shown us the frightening realities of modern-day surveillance technologies. With the automation of surveillance technologies, there are far more powerful possibilities available to unscrupulous individuals, organizations, and governments. Relevant intelligence information can be provided on diplomatic, economic, and scientific developments.

This presentation concerned what Campbell himself termed the "lawless area of the surveillance of international communications." Specifically, he reviewed the ECHELON project, a highly automated system for processing global communication information. Campbell uncovered this project for the first time in 1988. In 1999 he produced a European report with the first documentary evidence for the continued existence of this system and its capabilities. This extensive system exists to access, intercept, and process important global communications. Campbell suggested, at an interview during this conference, that the problem is that "it is clearly only too easy in this new environment for the rivers of information to be accessed once the portholes are open." He went on to add that now is the critical time to deal with these problems, "because if we don't get the right things . . . then our children and our children's children will turn round and say, "What was privacy?"

This presentation also reviewed the effectiveness of statutes for protecting privacy in the light of advances in automated surveillance technologies. As Campbell argues, "There is no law controlling the international snooping of communications." He presented evidence indicating that governments are routinely exploiting communication intelligence for commercial gain.

When asked prior to the talk what he believed attendees would leave the room remembering, Campbell suggested, "I think that what I have to say and show will take most people to places that they didn't dream existed." He added that he hoped he would be able to share with his audience the "scale and the capacity of international surveillance" and that this should "underscore what people should do in terms of organizational and self-protection in terms of leaching of information out of open communication systems."

PANEL: INTELLECTUAL PROPERTY AND THE DIGITAL ECONOMY

Summarized by Mathias Klang

Intellectual property is the hot issue of the digital economy. It stretches from theoretical arguments on the right to own information to serious everyday problems concerning the basis of Internet geography. The panel was described in the conference program: "New laws are proposed or adopted frequently to strengthen intellectual property rights. Contract and technical protections are strengthening intellectual property protection as well. This past year saw adoption of new trademark domain name cybersquatter legislation, significant developments in the legal protection for the contents of databases, approval of a new licensing law for computer information, and more legislation and case law on digital copyright issues, not to mention endorsement of e-commerce and business method patents that will have substantial impacts on computers, freedom and privacy. Some have even proposed giving individuals property rights in their personal information as a way to protect privacy."

With these points in mind, the panel attempted both to educate and to inspire the audience on what is both a legally complex issue and an important question for the CFP/IS community as a whole.

Yochai Benkler of the NYU Law School spoke on the topic of copyright, touching on the important issues of the new copyright directive in Europe and the two (almost incompatible) bills proposed in the U.S. One was the copyright in data, and the other concerned prohibition of duplication. Randall Davis's topic was "The Digital Dilemma: Intellectual Property in the Information Age." This discussed the collision between information and intellectual property. David Post led the audience on a legal tour of Internet geography by explaining the problems of trademark law in relation to the domain name system. The talk included the basics of trademarks, an explanation of the problem of consumer confusion on the Internet, and a discussion of the legal remedies open to the trademark holder. The talk ended with a critique of legal remedies: the courts can be economically inefficient and alternate dispute resolutions are not always enforceable. The conflicts involved in this issue guarantee that there will be plenty of work to be done in this area in the future.

Pamela Samuelson spoke on the basic foundations and purpose of intellectual property (IP) law. Starting with the

philosophical foundations of IP, she explained the rights granted the property owner and what these implied. She then continued with a discussion of the new set of problems created when IP law (especially copyright) is applied in a digital environment.

On the whole, the panel gave a well-rounded picture of the problem area, proving both the importance of sensible regulation (without attempting to define what that may be) and the need for legislation to be sensitive to the technological realities of the present rather than attempting simply to enforce the rules of the past.

PANEL: NEGOTIATING THE GLOBAL RATING AND FILTERING SYSTEM: OPPOSING VIEWS OF THE BERTELSMANN FOUNDATION'S SELF-REGULATION OF INTERNET CONTENT PROPOSAL

Summarized by Ernest Miller

In 1998, the Bertelsmann Foundation began a project whose overall mission was to facilitate the development of an integrated system for dealing with harmful and illegal content on the Internet through self-regulation.

In September 1999, the Bertelsmann Foundation issued a multi-part report on "Self-Regulation of Internet Content," addressing several issues, including the protection of vulnerable parties, finding and evaluating information, and detection of electronic crimes. Some of the recommendations of the report were seen as controversial when it was presented in Munich, and the controversy continued at the Computers, Freedom, and Privacy conference. [Editor's note: The author contributed to the Bertelsmann's report.]

Moderator Jean Camp began the panel by providing a comprehensive overview of the foundation's report. She noted that the report was framed as a series of questions, but that the panel would concentrate on only one – the answer to which was developed by Professor Jack Balkin of the Yale Law School, recommending a multi-layered approach to content labeling and filtering.

Panelist Dianne Martin from George Washington University provided context for the proposal by tracing the recent development of labeling and filtering systems, starting with RSAC and console-game ratings, through the development of W3C's Platform for Internet Content Selection (PICS), and ending with ICRA. Martin explained that the Bertelsmann proposal was a significant improvement over previous systems, because it separated the labeling function from the filtering function. More important, she noted that the proposal was both technically and socially more complex, permitting a larger context and multiple cultural value systems. The history lesson took a different turn as Christopher Hunter, Ph.D. candidate at the University of Pennsylvania, compared the system to the Catholic Church's list of banned books during the Middle Ages. Hunter feared that the system would not remain voluntary as the report recommended, but that governments would make compliance a legal requirement.

Nevertheless, he claimed, many sites would remain unrated and be banished to a "no man's land where browsers fear to tread." Jordan Kessler from the Anti-Defamation League supported the proposal and was pleased by many aspects of the system, including the choice it provides consumers of different red/green lists and templates, the use of encryption to prevent upstream filtering, and – most important – the default setting that unrated sites not be filtered. The role of the user was of key importance, argued Kessler. "Users are not sheep. If they are smart enough to turn the default settings off, they are smart enough to find and use white lists."

The final speaker was Barry Steinhardt of the American Civil Liberties Union, who listed a number of problems he found with the proposal. The biggest problem, claimed Steinhardt, was that Web sites would face a dilemma: either self-label and be blocked, or fail to rate and be blocked. He also saw the scheme as too burdensome for Web site creators, citing one artist's Web site with over 25,000 pages of content. Steinhardt also reiterated Hunter's point that the voluntary nature of the proposed system was illusory.

A number of questions from the audience revealed that the audience was as divided as the panelists on the issue.

THE DEMONIZATION OF PIRACY

Jessica Litman, Wayne State University Law School

Summarized by David Todd

The ownership of ideas and the control over their representation, both virtual and tangible, have long been manifest in the realm of intellectual property. Intellectual-property laws have typically been based on a number of assumptions, including some concerning the structural characteristics of the industries for which they are created to protect and assign rights. Embedded within these assumptions are the concepts of authorship, inventorship, distribution, and an implicit demarcation of territoriality. As businesses redefine and reorganize themselves to take advantage of computerization, the elimination of trade barriers, and the adoption of the Internet as a stable platform to build a new economy, traditional underlying assumptions surrounding intellectual-property issues must be called into question.

Jessica Litman challenges the assumptions of both the past and the present by problematizing the intellectual property regime in a global marketplace in which territoriality collapses and distribution is

no longer faced by the constraints of space and time. She understands current copyright law as the "demonization of unauthorized use," a fundamental shift from the copyright tradition of the past.

Two hundred years ago, copyright reserved a limited number of rights for a limited time to a limited number of authors. The logic of this was rational enough – controls held by authors were consciously structured to confine the author's rights while allocating enough control to provide for extraction of some commercial value. The purpose behind granting this type of copyright system was first and foremost to benefit the public at large and secondarily to ensure modest protections on the owner's work.

Over the course of some 200 years, however, the copyright system has changed on a fundamental level, and Jessica Litman notes that the United States is in the forefront of redefining these changes. Over the course of the past century, copyright law has evolved from a quid pro quo framework into a model that is rooted in supplying an economic incentive to distribute works. This established a direct relationship between copyright extension and authorship. The principle of fair use thus established its paramountcy, as authors and publishers alike required a level of protection as an incentive to produce works.

Over the past five years or so, copyright has shifted away from an incentive model to an issue of control. Control, however, requires the assignment of power over the uses of a product. Thus, for example, the RAM copy issue and the copying of MP3 files represent a watershed moment in the development of copyright control. Here, anything that might resemble the "effect" of piracy is deemed to be in fact an act of piracy. The industries that have a vested interest in copyright protection now interpret unauthorized use as theft, an act of piracy. The negative connotations of theft and piracy resonate, and the "demonization of unauthorized use" pervades the social, political, and legal arenas. This has been manifest by a call for action to obtain extensive legal protection against behavior that was deemed legitimate only a decade ago, when control by copyright owners over some uses while leaving others to the public was considered a desirable feature of the copyright legal scheme.

How do we work within the parameters of this emerging control regime, where the cyberpirate/unauthorized user is seen as the evil other? Jessica Litman offers two possible solutions: Attempt to persuade the courts to read the law in a way that is more reasonable, or, failing that, engage in widespread noncompliance of the law that has transformed unauthorized use into the demonization of piracy.

PANEL: INFOMEDIARIES AND NEGOTIATED PRIVACY

Summarized by Mark Hissink Muller

Many people came to this session to hear about these initiatives to help the consumer get a grip on his or her privacy. Moderator Jason Catlett indicated that the topics would be addressed one at a time, starting with infomediaries.

An infomediary is probably best described as a broker for consumer information. Currently a lot of money is made by certain companies selling people's personal information. Infomediaries base their activities on the idea of giving consumers a chance to benefit financially from their personal information being made available to third parties.

Many consumers won't have the time to surf the Net extensively to find the best deal for a product. Infomediaries think people are interested in having that work done for them. Some consumers may be interested so long as their privacy is well protected.

Catlett started by asking the panel whether the rise of the infomediary is a good thing for privacy. Privacy advocate Beth Givens stated that the key issue is trust, and she made a few critical remarks. She stated that it is not a good idea to trust a company with much personal information, because companies are just not stable enough and tend to change their policies. In a rapidly changing environment, it's not possible to predict how a company will handle your specific interest in a few years. Criticaster Alexander Dix added that the business model of the infomediary is not economically viable if they are just guarding consumer's personal information, and that sooner or later they would have to resort to selling secondary information to "untrusted" third parties. As they would start combining online and offline information, consumer privacy would diminish. Although he did like the idea of the infomediary in theory, he said, there are just too many practical problems.

After that, the floor was given to the representatives of various companies that more or less could be seen as infomediaries. First, Steve Lucas from Privaseek said that his company was founded two years ago with the vision of giving consumers the ability to "own" their personal information. Steve tried to take away the privacy advocate's fear by stating that Privaseek allows consumers to access and control what they put in the database. Apart from that, Privaseek also maintains a very strict policy as to which companies they share data with. Their partners are audited and bound to strict regulations.

AllAdvantage's representative Ray Everett-Church explained that their business model differs in some ways. First of all, AllAdvantage does not trust third parties to utilize customer information. Because their partners have no direct access to the information, they are not able to use customer information.

Second, AllAdvantage uses a viewbar to interact with customers, so the customers can turn the viewbar off when they do not want their browsing to be monitored. Everett-Church declared that when a customer decides to quit membership, all entries are deleted.

Paul Perry from Microsoft explained that their Passport technology enables a user to log on just once per session, after which a consumer does not have to type a password when visiting a Web site that uses Passport technology. Perry stated that the information given to the Web site that is visited is very limited, so Passport is not an infomediary but functions more like a mega-service to the consumer. He stated, however, that Passport is a prerequisite for infomediaries.

Members of the panel indicated that infomediaries would not make legislation unnecessary. As Everett-Church put it, "Legislation is necessary for creating an equal playing field from which infomediaries can deliver." Dix added that right now infomediaries policy is aimed at collecting as much information as possible, whereas the policy should be aimed at minimizing data collection.

P3P is intended to be a technology that enables the user to exercise control over his or her personal privacy policy. This could be implemented by sounding an alarm bell when a company's privacy practice differs from a consumer's preference. In today's Web surfing there is just the SSL-icon to indicate a secure connection. There is no indication whatsoever of a Web site's privacy statement.

P3P has been heralded as the savior of privacy. Industry can use it to say, "We don't need legislation, we've got P3P." Alexander Dix applauded the goal of P3P, but stated that the society-neutral framework would have to be filled for every country by law. "P3P technology is a tool; it's necessary but not sufficient." Moderator Catlett pointed out that P3P assumes companies use a specific policy, but in fact their actual policy can differ from the P3P setting.

The main criticism of P3P was that nothing has been delivered to date, and it is questionable whether companies would actually be interested in adopting the technology. Is there an incentive? Everett-Church indicated that companies are not likely to install a technology that limits the user-friendliness of their Web sites. You need carrots if you want to encourage companies to adopt; sticks are not going to work.

Beth Givens added to this that she's happy with the concept of P3P, but really people must be educated to acknowledge the value of privacy. "That's why I'm so very happy with CFP2000."

PANEL: CIRCUMVENTION: TOOL FOR FREEDOM OR CRIME?

Summarized by Brett Burney

Is circumvention a tool for freedom or crime? This session discovered that answers to this question are few and far between.

The session was well attended and the panel (Robin Gross, Declan McCullagh, Paul Schwartz, Barry Steinhardt, and organizer Alex Fowler) expanded to include John Gilmore, Pam Samuelson, and Jessica Litman. Questions from the floor came from lawyers, techies, and a librarian.

Fowler started the discussion by asking Gross, staff counsel for the Electronic Frontier Foundation, to discuss her work on the DVD/DeCSS case. She explained that two of the four cases involving the DeCSS utility hinged on the new Digital Millennium Copyright Act (DMCA). There are two main sections of the DMCA under fire in these cases: (1) the act of circumvention, and (2) the creation and distribution of tools that allow the act of circumvention. Panel members pointed out that the second section is no longer an intellectual-property issue. Rather, it is a free-expression issue, given that it is a general prohibition on distribution. However, Gross pointed out that the question remains one of intellectual property simply because there is a potential for copyright infringement.

Barry Steinhardt of the American Civil Liberties Union spoke on the CyberPatrol case. The "tool" in this case was a little program called cphack that (1) allowed users to unmask the list of sites that were blocked by the CyberPatrol program and (2) allowed users to discover hidden passwords. Schwartz pointed out that the significance of this case lies in that it did not involve piracy or the obtaining of illegal copies of CyberPatrol but, rather, the circumvention of certain protections.

Paul Schwartz, an expert on privacy from the Brooklyn Law School, talked about the "quasi-public privacy exception" section of the DMCA. This subsection apparently allows users to circumvent technologies in order to protect their privacy. Pam Samuelson pointed out that if this subsection does indeed allow users to circumvent, other parts of the DMCA prohibit you from creating a tool to allow you to do the circumventing! Samuelson continued that the DMCA appears to be "rife with contradiction" and "totally incoherent." Taking her argument, one could read into the DMCA an embedded right to create the tools, but then a person would still be prohibited from the actual distribution of such tools.

Lastly, Declan McCullagh from *Wired News* spoke briefly on the difficulty of getting the general public interested in cases involving intellectual property. While he did say that it was possible to get an editor interested in such a story by using creative and catchy headlines, he argued that the general public is not yet motivated enough to get involved in

issues and cases that will affect it more than it realizes.

The session was very enlightening and interesting. Several questions defined difficult contemporary problems that the panel could not answer, simply because there was no answer. Samuelson noted that she had spoken with several writers of the DMCA who told her they had purposely left sections ambiguous so that they could be worked out later in the courts. We can only hope that the right cases land in the right courts and that the question of "freedom or crime" is answered correctly.

PANEL: HUMAN SUBJECTS RESEARCH IN CYBERSPACE

Summarized by Kurt M. Saunders

The Internet has made possible new avenues of research involving human subjects. Traditional academic research involving human subjects is governed by ethical standards and laws designed to protect the privacy and anonymity of the individuals serving as research subjects. For instance, an independent oversight board reviews the design of the proposed research. Likewise, proper informed consent by the subjects after an explanation of the research and an opportunity to ask questions is an integral part of the process. Do these guidelines apply in the virtual world to protect human subjects and yet allow for scientifically sound research?

The session was moderated by Professor Bruce Umbaugh, a philosopher and director of the Center for Practical and Interdisciplinary Ethics at Webster University in St. Louis. The first panelist, Sanyin Siang, co-author of the American Association for the Advancement of Science report on human subjects research in cyberspace, provided an overview of the standards that govern human-subjects research. She explained that such research rests on four principles: respecting subjects as autonomous individuals who must give their informed consent; maximizing possible benefits for the subjects; protecting their privacy and confidentiality; and fairly distributing the burdens and risks associated with the research. According to Siang, applying these principles to research in cyberspace is difficult because the distinction between the private and public domains is blurred and because the use of anonymity and pseudonymity may obscure issues of identity.

Panelist Amy Bruckman, an assistant professor of computer science at Georgia Tech and founder of MediaMOO and MOOse Crossing, spoke next. Using the example of doing research by lurking in a chatroom to observe, record, and analyze the chats, Bruckman considered the use of analogy and genre to define possible ethical issues. Is lurking like sitting on a bench in the town square and listening to the conversations of passers-by? Is logging a chat like taking notes? Are pseudonyms the same as real names for the purposes of the chat? According to Bruckman, such analogies are both powerful and dangerous, because the chatters' reasonable expectations of privacy are critical. Instead of analogies, she urged that the members of the online community develop and evolve their own standards that will form a reasonable expectation of privacy.

Next, panelist Julian Dibbell, author of *My Tiny Life: Crime and Passion in a Virtual World*, which concerns his experiences on LambdaMOO and the "LambdaMOO Rape" incident, considered the topic from the standpoint of journalism ethics. He noted that the ethics of journalism differ from those of scientific research, in that ethical decisions are often made ad hoc, subject to less rigorous and less institutionalized standards. For instance, he had initially decided to print entire postings from LambdaMOO; however, some members asserted that they had a copyright in their postings and that his copying of the entire passage did not qualify as fair use. In effect, they resorted to copyright law to preserve their confidentiality. Dibbell concluded that the subjects' reasonable expectations of privacy must be determined on the basis of the context and nature of the subject researched.

Finally, Bruce Umbaugh added that many of these problems stem from the insecure nature of the medium itself. Consequently, he defined two principles that should apply in online human-subject research. The first is the principle of equality, whereby all subjects should be treated in the same way. The second principle is that of individual control, whereby all subjects should be able to reserve control over their private information.

PANEL: NETWORK SOCIETY AS SEEN BY TWO EUROPEAN UNDERDOGS

Summarized by Alma Whitten

Internet use in Italy and Spain is growing, but both countries are still well behind most of Europe in their proportion of number of Internet hosts to population size and economy. This session examined some factors that have affected the development of Internet use in those countries, sometimes in surprising ways.

Giancarlo Livraghi began by arguing that the nature of the Internet is biological and ecological, distributed and without a center, and that this is also true of Internet culture. Because of this, we cannot look at the development of Internet usage in different countries as simply different stages of the same process; instead, we must recognize that different cultures must follow different paths.

Andrea Monti spoke about the notorious 1994 Italian "Crackdown" in which many homes were raided and computers seized in a hunt for illegally copied software, explaining that the police were at that time technologically unsophisticated and did not know to draw a distinction

between seizing data and seizing the medium on which the data resides, and that computer seizure was not yet recognized as a human rights issue. Ironically, Italian law actually requires that users have the right to make a backup copy – a point often conveniently forgotten by corporations. The copyright lobbyists in Italy also want to prohibit and criminalize any exchange of information about how cryptographic intellectual-property-protection mechanisms work.

David Casacuberta then talked about several interesting incidents in Spain, beginning with the Telefónica boycott, in which early Internet users were brought together in a fight for more reasonable rates. Initially, politicians attempted to portray the Internet users as a greedy elite who wanted everyone else to pay higher phone rates to subsidize their Internet hobby. The media initially concurred, but eventually switched sides and became sympathetic to the boycott. In the end, the struggle for a flat rate contributed to the development of a stronger Internet community, which then went on to tackle more philosophical issues such as freedom of expression.

After describing the boycott and its effects, Casacuberta went on to characterize DNS services in Spain as "chaotic, corrupt, and stupid," whereupon Livraghi and Andrea Monti added that in Italy the DNS management is chaotic and stupid but, as far as they know, not corrupt. Spanish DNS rules require that domain names be at least four characters long and not consist of common words, but powerful corporations get domain names that violate those rules. David next described an incident in which an ISP that hosted a Basque nationalist Web page was subjected to severe email bombing after a horrible terrorist incident that had only tenuous connections to the Web-page authors.

The media covered the mail bombing without condemning it and in their reporting published the email address of the ISP, which led to a redoubling of the bombing. Opinions still differ on whether this was an innocent mistake or done under direction from the government. Afterward, there was much discussion of the ethical considerations of the situation. When a similar situation arose at a later date, no mail bombing occurred, because the prior discussion had engendered a new respect for free expression. Finally, Casacuberta discussed another Web page, authored by an anti-torture group, which lists the names of police who have been on trial for brutality and torture. This is not private information per se, since it is available in the newspapers, but when a new privacy law was passed, action was taken against this Web page as a "privacy-violating database." The page was taken down, but mirrors of it were then put up, and the argument continues.

PANEL: "WHO AM I AND WHO SAYS SO?" PRIVACY AND CONSUMER ISSUES IN AUTHENTICATION

Summarized by Sarah Wilford

Under the moderation of Deirdre Mulligan from the Center for Democracy, panelists Margot Freeman Saunders, managing attorney for the National Consumer Law Center, Carl Ellison of Intel, Phil Hester, vice president for systems and technology at IBM, and David Flaherty, of the University of Victoria, discussed the hot topic of authentication and electronic commerce. In discussing the problem of verification of identity for medical records, the panelists said that trust and reputation are key points in the use of verification. The potential for abuse of key data by certification registers was considered, along with the responsibility of the producers of authentication technology to ensure its security.

Looking forward, the panelists stressed the need for clear understanding not only by academics but by all members of society in order that trust in certification organizations and the security of the data banks they maintain be assured. The problem of identification of individuals of similar or the same name was also raised. This issue is particularly evident in online communities, which may have thousands or even millions of participants.

Perhaps one of the most difficult areas is the proper compromise between the need for authentication and the desire to maintain the privacy of the individual. This was seen as an area in need of much further research.

DEBATE: INTERNET VOTING: SPURRING OR CORRUPTING DEMOCRACY?

Summarized by Kat Hanna

Internet voting – it can provide convenient access to democracy, unfettered by transportation difficulties or scheduling conflicts. It has the potential to make it easier for people with disabilities to participate in the political process. It could well transform democracy. But will it be a positive transformation? Or will it make it easier for the relatively affluent to participate in the electoral process while further marginalizing those who are less well off? Will the lack of in-person authentication and oversight make coercion and fraud more widespread? Online voting is subject to the pitfalls associated with any activity conducted over an inherently insecure network. Can we make online elections secure? And will the public trust the integrity of such elections? These issues were the subject of a lively debate among panelists of varied backgrounds and opinions. The voices urging caution far outnumbered the one saying we're ready for wide deployment of online voting.

Moderator Lance Hoffman of the George Washington University began with a brief introduction of the topic. He out-

lined the history of remote voting, noting that paper-based absentee ballots have long been used in government elections in the U.S. In addition, online, legally binding elections have been held by labor unions, trade associations, nonprofit organizations, and other private-sector groups. Hoffman pointed out a few recent experiments in political elections over the Internet. In January of this year, thirty-five participants in the Alaskan Republican Party's straw poll voted online. In March, online voting was available to all participants in the Arizona Democratic primary.

Panelist Joe Mohen of election.com began the debate. Election.com ran the Arizona election, and Mohen went into greater detail on that event. He reported a significant increase in voter turnout over previous years, especially among Native Americans and African-Americans. Mohen further argued that security issues for online voting have been addressed, asserting that the existence of adequate security for online banking and stock trading implies sufficient security for online elections.

Hans von Spakovsky of the Voting Integrity Project spoke next, urging caution in the rush to wire the ballot box. Von Spakovsky noted three conditions necessary for fair and free elections: equal access, ballot secrecy, and ballot sanctity. Making it easier for a certain segment of the population, such as relatively affluent Net users, to vote is not only unfair but illegal. Von Spakovsky also expressed skepticism that security concerns have been adequately addressed.

Next the audience heard from David Jefferson, chair of the technical committee of the California Secretary of State's Internet Voting Task Force. He described a study conducted by the Task Force that was begun with a great deal of enthusiasm but ended by recommending great caution and much further deliberation. Jefferson characterized the actual and apparent security of the voting process as nothing less than "a national-security issue." He went on to enumerate several problems he sees with the Arizona election, including the possibility of corruption by viruses or trojan horses, weak authentication of participants, the potential for denial-of-service attacks, and discrimination based on operating system and browser.

Paul Craft is project director of the Florida Department of State, Division of Elections, Voting Systems Section's Internet Voting Initiative. In Craft's view, the move to online voting is coming but should be approached with a cautious, evolutionary attitude. Craft expressed concern regarding the decision of the Arizona Democratic Party to move the primary online and wondered how the decision was made.

Barry Schoenmakers of Eindhoven University in the Netherlands approached the issue from a more technical perspective. He argued that a major stumbling block to the adoption of online voting is lack of transparency of the process. In paper-based elections it's fairly obvious how to determine potential fraud. This is not the case with Internet voting, so we need a technical means by which to compensate for this opaqueness. Schoenmakers outlined an approach based on homomorphic encryption which allows both the decoupling of digital signatures from encrypted ballots and the ability to tally and verify the still-encrypted votes.

The presentations were followed by a heated debate among the panelists, who discussed statistics from the Arizona election, issues of online voter registration and authentication, and security concerns for both clients and servers. The audience eagerly weighed in with questions addressing the privacy of votes with regard to the vendor running the election, use of information gathered by vendor, and the possible corruption of the neutrality of the vendor.

CFP2000 HOT TOPICS: HEALTH PRIVACY

Summarized by Lina Tilman

"Do not assume that no one knows about your health but you," warned Ari Schwartz, the moderator of the Health Privacy panel and a CDT policy analyst. Healthcare consumers' lack of privacy and control over their medical data in the digital world has inspired debate and controversy for decades; today, medical records remain the most widely circulated of all records distributed to third parties. Certain privacy restrictions apply to healthcare providers, clearinghouse employees, and healthcare plan workers; however, access, distribution, and use of medical data by researchers, public-health workers, law-enforcement officials, and members of the press are virtually unregulated. "Has the battle already been lost?" Schwartz asked the diverse, albeit uniformly American, panel.

Peter Swire, the chief privacy counselor of the U.S. Office of Management and Budget, expressed optimism regarding the government's and the industry's ability to "build privacy into [information] structures." Swire outlined the U.S. legislative history of bills, proposals, and initiatives that addressed medical privacy, noting that Congress has made substantial progress in the 1999–2000 period. Swire's data indicated U.S. healthcare consumers' serious engagement and concern with regard to their health privacy: 60,000 comments, dozens of them consisting of a hundred or more pages, were submitted in response to the 1996 HIPA (Health Insurance Portability Act), which required Congress to address health-privacy issues. Arguments in favor of strict regulation of healthcare information include economic ones, rights-based ones, and those founded on public concern. "Records save lives," Swire conclud-

ed. "We want [consolidated medical information], but with better privacy."

Angela Choy, a field director for the Georgetown Health Privacy Project, rejected the validity of the common perception of a conflict between consumers' privacy and practitioners' effective access to medical data. Choy argued that greater privacy leads to improved healthcare research and service, whereas lack of privacy causes healthcare consumers to interfere with and impede the quality of their healthcare. Fear of disclosure (hence, fear of stigmatization, loss of employment, loss of insurance, etc.) may ultimately lead to avoidance of medical aid altogether. Protection of privacy thus signifies protection of public health.

Greg Miller, chief Internet strategist of MedicaLogic.com, described a variety of health risks, inefficiencies, and instances of malpractice that could result from unavailability of consolidated medical data. MedicaLogic.com, through integration and centralization of patients' health records and workflow reengineering, creates digital health portfolios that seek to reduce errors and omissions, thus enhancing healthcare quality while reducing costs. Miller acknowledged the issues of security, privacy, and reliability as the primary concerns of MedicaLogic.com. Rebecca Daugherty, representing the Reporters Committee for the Freedom of the Press, argued that health-privacy regulations should not cut off reporters' access to truthful, nonstigmatizing information; privacy regulation should not penalize whistleblowers and cripple reporters. Daugherty claimed the existence of strong public interest with regard to health and healthcare issues. "The public must know," she stated, "if an airplane pilot has had an alcoholism problem." In response to Daugherty, information and privacy commissioner of Ontario Ann Cavourkian expressed concern regarding backdoor disclosure of healthcare records and unscrupulous reporting to cater to public curiosity.

On the whole, the panel appeared to acknowledge the legitimacy of healthcare consumers' increasing privacy concerns and to agree that even those interests of the press that represent legitimate public interest must be balanced against consumers' right to privacy.

PANEL: IS TECHNOLOGY NEUTRAL? SPACE, TIME, AND THE BIASES OF COMMUNICATION

Summarized by Megan McCormick

The panel began with a statement by Reg Whitaker, professor of political science and an author on privacy issues. Whitaker argued that surveillance technology has increased in prevalence throughout the twentieth century, and that this growth was marked by a decrease in state power over its citizens. The power made possible by an increase in surveillance technologies has been falling more and more into the hands of corporations, leading to what Whitaker called a loss of citizenship, replaced by a growth of "consumership." These economic powers do not coerce consumers but, rather, have set up a structure of consent whereby people willingly allow their privacy to be eroded. In order to counter this development, Whitaker suggested that rather than consider privacy an individual right, it should be refigured as a social right that serves a public good. In this way, some of the negative side effects of willing loss of privacy by individuals might better be addressed as dangers to the community.

Community was also at the core of the remarks of the second speaker, Marita Moll, educator and member of the Canadian Teacher's Federation. It is a popular belief in education, Moll said, that computers with Internet capabilities must be brought into the classroom as quickly as possible. This, according to supporters, is critical in order to provide children with all the opportunities that such technology brings.

However, Moll says, the forces that drive this push seem less concerned with the educational needs of the children than with the needs of the businesses that support and supply this technological training. This is brought clearly to light when one considers that, in Canada, the push to bring such technology to the classroom falls under the authority of Industry Canada (the Canadian government's equivalent of the U.S. Commerce Department) rather than Departments of Education. [Editor's note: in Canada, there is no Department of Education at the federal level.]

In addition, there are negative side effects to bringing technology into the classroom, especially technology that, Moll says, has a very "globalizing" effect. Introducing schoolchildren to the Internet may have the effect of homogenizing education, removing traditional cultural distinctions among different areas, and undermining the community structure of education.

The final speaker, author Paulina Borsook, addressed the ideology of the high-tech industry itself. Borsook claims the industry has a certain image of virtuality – virtual workplaces, virtual communities – that it does not match in practice. No matter how much the high-tech industry may claim that new methods of commuting made possible by technology are changing the face of work, workers in the industry know that in order to be taken seriously they must be with other technology companies, in specific geographic locations. In addition, the rhetoric of the high-tech industry ignores the actual resources and costs of the industry, be they the way cities change to support certain types of workers or the way families are affected by the demands of high-tech work. To Borsook, the industry suffers from a gap between rhetoric and reality that is consciously ignored, and that has real-world consequences that are also dismissed.

Audience response to the panel was mixed. Having been presented with these many negatives of technology, said one woman, how should the audience proceed? Are these negative effects an inevitable result of technological development, or, as one man asked, was there another way things could have gone? The remainder of the discussion considered these questions, as both panelists and audience members struggled with the themes of balance between the advances made possible by technology and their consequences, which are too often unnoticed until too late.

PANEL: INDIRECT THREATS TO FREEDOM AND PRIVACY: GOVERNANCE OF THE INTERNET

Summarized by Thomas Nauer

The architecture of cyberspace is based on technical standards that essentially define our world. Thus standardization is in some sense governance. In this session an attempt was made to define and map this governance, or at least the architecture that constitutes Internet governance. The three panelists are all deeply involved in the standards process: Timothy Schoechle is at the International Center for Standards Research, University of Colorado, Fred Baker is chair of the Internet Engineering Task Force (IETF), and Jean-François Abramatic is chair of the World Wide Web Consortium (W3C).

For Schoechle, there is a basic conflict in two major approaches to governance: the use of law, and the use of self-regulation. The latter is common in the U.S., while the former is more preferred within the European Union. The actual governance of the Internet is hiding in the form of technical architectures that are defined by standards bodies. They are dominated by private entrepreneurs, by corporations, and by groups of people who get together and make technical standards. Schoechle's thesis is that the voluntary consensus standards process could contribute to solving broader governance issues. The underlying concept is the notion of the "public sphere."

Currently, discourse on policy is about either the public sector (government) or the private sector (individuals and private companies); this dichotomy omits the most important element, namely, the public sphere (as described by the German political philosopher Jürgen Habermas), which overlaps both of them. It emerged from ideas generated in public discourse in coffee houses during the seventeenth and eighteenth centuries, which solidified the concepts upon which our modern society is based. The public sphere consists of private parties gathering in a public place to discuss ideas of mutual interest, completely apart from government. This notion serves as an excellent definition of a standards committee. Yet some difficulties remain, e.g., how to involve more people, such as users and the general public, in the process. Standards bodies often comprise technical people from corporations, rather than members of the general public.

Organizations such as the IETF designed most of the infrastructure protocols that are used on the Internet, wherever it might be. Or, as Fred Baker sees it, "The Internet is not international, it's a-national: it ignores the existence of nations." It is only recently that the IETF became involved in broader governance issues, e.g., in the context of the so-called "Raven process," which dealt with privacy issues raised when engineers from various companies started working on a variety of gateway controllers to insert wiretaps into the network. But would this provide solutions to subvert other countries' or competitors' networks? Privacy and democracy are not issues the IETF wants to work on, but doing something that does not promote them turns the Internet into a technology people cannot or should not use.

Abramatic discussed the term "governance." In his understanding it has to do with French-style centralized power rather than the decentralized structure of the Internet society. He also counters allegations that the W3C is a closed and opaque organization. The members of W3C are companies, not people, like the IETF; therefore the structure appears less clear. And, like the Information Society, the W3C is based on networks rather than hierarchy. This could be illustrated with its four different domains of activity (User Interface Domain, Technology and Society Domain, Architecture Domain, and Web Accessibility Initiative) and its recognition of the international nature of its work. Hence, Abramatic preferred to think of the World Wide Web Consortium as a meeting ground for huge challenges in terms of technology. He said there is no chance for success if communities concerned with its decisions are not involved. And to achieve this, there is a long way to go.

PANEL: PERSONAL DATA PRIVACY IN THE PACIFIC RIM: A REAL POSSIBILITY?

Summarized by Nadia Olivero

Conforming to expectations, the Friday afternoon session on Privacy in the Pacific Rim raised important issues for the development and applicability of privacy policies. As was suggested by the session subtitle "A Real Possibility?" the panel discussion underlined the problems related to the coexistence of different cultural and social systems and different approaches to privacy regulation. Professor Jim Tam from Ryerson Polytechnic University moderated the session and pointed out the necessity of exploring the state of privacy affairs in regions such as the Asia Pacific Rim, where there is advanced use of technology but conflicting political and cultural situations. Differences in trends of technology adoption seem to correspond to

differences in political development and human-rights awareness.

Whereas the concept of privacy is generally accepted in the West, it is a new concept to many Asia Pacific countries and needs to be promoted at the same time that it is being protected. Today, those countries that are oriented toward a model of control are more likely to adopt new surveillance technologies. Jim Tam reports wide violations of privacy through surveillance of messages and interception of communication between citizens. The government of Singapore watched 200,000 computer users during the pretest of a scan for a virus, while in China companies and individuals are normally "requested" (read: required) to disclose personal information.

In this regard Stephen Lau, Privacy Commissioner for Personal Data in Hong Kong, gave an account of the increasing significance of privacy as a basic right in China. A 1999 "Community Opinion" survey asked respondents in Hong Kong to rank the importance of social-policy issues. The results showed that the population claims privacy as a significant and fundamental right. The increasing importance of privacy in conjunction with growing Internet connectivity in the region leads to important economic and social consequences that need to be addressed promptly. Given these recent developments, which regulatory model should be adopted and how will people react to and implement privacy regulation?

The session stimulated reflections on the existing models and compared the different modus operandi between countries. Some countries, such as Singapore and Malaysia, which have little or no Internet regulation argue that privacy would be protected by sector laws. Other countries, such as Japan, Korea, Thailand, and Australia, have laws that cover the public sector but not the private sector, and thus are only partially regulated. Still others, like China, have some laws for email privacy, but little else. Senator Kate Lundy provided a detailed picture of the current bill in Australia and its next probable developments. The current bill, based on a co-regulatory model, seeks to extend privacy regulation into the private sector. Despite promises, it is still pending in Parliament after having been withdrawn in 1997 by the then liberal government.

Since then, after a change in government, privacy legislation has not yet been properly addressed. The bill itself presents some deficiencies. Based on a co-regulatory model aiming to conform to requirements of the EU Data Protection Directive, it should guarantee regulation and control by the Privacy Commissioner. Nevertheless it seems that in the last few years its funding and potential impact have been progressively limited. The underlined picture shows that in Australia as well there is a need to improve and develop actual legislation. There is the need to build an environment that is genuinely co-regulatory, in which all the parties involved could participate in shaping the new assumptions of privacy protection and privacy rights.

Beside legislative efforts, other realities show the need to educate the population about the right of privacy. As was effectively pointed out by Professor Jim Lin from the National Central University, Taiwan, privacy is a fundamental issue for the development of democracy. The low level of public awareness and concern about privacy in Taiwan was recently demonstrated when the government presented the project of an ID smartcard which was to contain all possible personal information, including financial personal data. Surprisingly, the population agreed to the idea without showing any particular concern. The proposal was scrapped after interventions from the relatively small "privacy community" in Taiwan, but it underscores the fact that privacy is sometimes not seen as a fundamental right.

In Professor Lin's view, despite the existence of an adequate legislative framework, there is the need to promote cultural changes with regard to privacy awareness. This view came out again in the discussion, with the general consensus that under any implementation model the attitude of the parties involved is crucial to its success. The cultural heterogeneity between countries would then require tailored interventions, which could take adequate account of the different perceptions and attitudes toward privacy.

In a region undergoing rapid political and social change, it is tempting to question whether increasing the importance of privacy is a matter of increasing democracy first or whether privacy awareness would be part of a dynamic evolutionary process toward more democracy. Moreover, even when democracy is the status quo, privacy protection depends on the active participation of all the parties involved, with particular reference to the private sector.

Finally, from the audience some more question marks emerged: under which conditions and circumstances will the private sector be more likely to support a data privacy regulation? Which model of regulation would be the best? And, given the ideological supremacy of the co-regulation pattern, which measures are required to make it work? For these questions there are no easy answers. Nevertheless, as was evident from the session, while the fact of international electronic commerce and information exchange does not require a single uniform approach to protection of privacy, it does require agreement on the importance of the concept.

PANEL: 10 YEARS OF CFP: LOOKING BACK, LOOKING FORWARD

Summarized by Alexander Macgillivray

Larry Abramson, panel moderator and telecommunications correspondent for National Public Radio, opened the "Looking Back, Looking Forward" panel with a challenge to consider how CFP would go into the new century with the freedoms it had gained while instilling strong ethical values in the cyber-citizens of tomorrow. The panel and the audience took his challenge and examined the interplay between freedom and responsibility in the frank and sometimes tense hour-and-a-half of discussion that followed. Barbara Simons, president of the Association for Computing Machinery, pointed out, "Ethics are a Pandora's box: once you open it, it brings all sorts of problems." Even though "our community is very good at denial" because "people don't want to know about things that are difficult," Simons said that it is important to face the ethical implications of one's work, to be just as interested in who is funding the work and how it will be used as in exploring the technical issues.

Stewart Baker, a lawyer at Steptoe & Johnson in Washington, D.C., and formerly general counsel to the National Security Agency, lamented what he saw as the disproportionate focus of this year's conference on Computers and Privacy to the detriment of Freedom. Baker thought that CFP had made some good progress on privacy, and he pointed to the Microsoft Universal ID Number and Intel Serial Number incidents as signposts of that victory. In both cases he said the U.S. government had brought pressure on companies invading privacy and this intervention had been successful in changing the privacy-invasive policies.

As another panelist, Ron Plesser, a lawyer at Piper, Marbury, Rudnick & Wolfe in Washington, D.C., told the audience, ten years ago the question was, "Will e-communications be regulated?" Now, because of the ubiquity of use of the Internet for business and pleasure, the only question is, "Who will regulate the Internet and in what manner?" According to Ben Smilowitz, a political activist and the panel's representative from the next generation of cyber-citizens, his generation may be ill equipped for the task. He lamented the fact that, though his generation is tremendously well equipped to handle the Computers of CFP, many in his generation, particularly computer-science majors, do not care about politics and so may be left out of the Privacy and Freedom policy debates.

Simon Davies, founder and director of Privacy International, stressed that this year's CFP even more than others was too quiet to grab attention. Davies still sees the Computers, Freedom, and Privacy landscape as an "us" versus "them" battle: "those with vision, and the parasites." He said, "Over the past ten years, the parasites have grown in number, wealth, and influence," so that there is "a rancid stench" in the air, and though "we should be extremely angry we are calm." His only hope is that "a small but growing number of companies have developed an ethical compass."

Ron Plesser, who identified himself as Davies's "them," countered that all of the sides in the debate over the future of technology must "get to us and get to yes on these issues." He pointed to industry, activist, and government cooperation on H.R. 3783, the Child Online Protection Act (COPA), as a good example of what can come from such collaboration. He hoped in the coming years similar collaborations could impart a value system for the Internet that all parties could be happy with.

Plesser also commented that CFP has always fostered open debate among the wide range of conference participants, but more than one audience member questioned the panel and CFP 2000 organizer Lorrie Cranor, who stepped in to answer some of their questions on behalf of CFP, about why women and minorities were so poorly represented in the audience and in speakers' list. Davies responded that the makeup of CFP was a reflection of the relative interests of different societal groups in the topics of the conference. Baker jokingly explained the overrepresentation of men in privacy circles as being because "men have more to hide or we need more help to hide it." But Simons granted the audience that it was a difficult problem and one that the ACM and CFP have tried, and must continue to try, to fix. The difficulty, Davies summarized, is that activist organizations like CFP are overworked and underfunded and they need to expend a lot of energy to do effective affirmative action. However, all agreed that the digital divide may be the pressing issue for CFP in the coming years and that CFP must continue to find the energy to attempt to address the problems maintaining the divide as well as it can.

Jessica Litman, a professor of law at Wayne State University and Thursday's lunch speaker, returned to the question of what ethics to leave our children by speaking about her own experience with her children. She acknowledged the "perfectly natural impulse" to want to keep control but cautioned conference attendees that much of what is evil in the laws has been motivated by the destructive desire to keep control in a changing world.

Instead, Litman argued that we should pay attention to the architectures we are erecting but remember that our children will build their own world and create their own set of values. She said, "It is not our job to design what tomorrow's world looks like. The best we can do is try to keep it safe for people like us – if our children are like us – without trying to make it unsafe for people who are not like us."

smart-space researchers

An Interview with Kevin Mills and Alden Dima

by Kevin Mills

<kmills@nist.gov>

and Alden Dima

<alden.dima@nist.gov>

Editor's note: Rik Farrow conducted this interview electronically. Kevin Mills and Alden Dima, researchers at NIST, are working on Smart Spaces and Jini-enabled devices.

Rik: Would you describe in some detail what you are doing at NIST?

Kevin and Alden: In a recent year, 1997 I believe, 4.2 billion microprocessors shipped. Only about 165 million of these microprocessors shipped in desktop personal computers – the remainder shipped as microcontrollers embedded in devices, such as microwave ovens, automobiles, televisions, and climate-control systems. Almost all of these embedded microcontrollers consisted of standalone computing applications. Yet, when you look at where future technology is leading us, microprocessors are rapidly moving to integrate most functions of an entire computer system on the die. Similar miniaturization in radio technology is also leading toward integration of radios onto chips. Industry's Bluetooth initiative provides one example of that trend. Given these trends, we can foresee that the embedded-microprocessor market will become an embedded system-on-a-chip (SoC) market, and that many of those SoCs will contain integrated wireless radios. So, in some future year, we might see four billion SoCs shipping with the capability of networking with each other. Many of these SoCs will be embedded in our environment, and many will be embedded in mobile devices that we will carry around from place to place.

Several networking and software issues may inhibit the development of the marketplace for embedded SoCs. Primary among them is automatic configuration and dynamic discovery. We cannot rely on human system administrators to configure and administer four billion computer systems, many of which will move from place to place. In addition, most mobile devices will be designed to leverage services available on larger processors available through more traditional wired networking infrastructures. To leverage such services, embedded SoCs will have to discover their presence, gain authorized access to them, and then configure themselves to exploit them. Other issues arise because developers of software applications will have to use new models for programming applications in such dynamic environments. And, of course, user interfaces will become distributed across a range of devices that must cooperate to provide humans with a seamless and effective interface. All of these issues are challenging, and there are folks here in the NIST Information Technology Laboratory working on each of these topics. Alden and I are focusing most directly on issues related to dynamic discovery, automated configuration, and new programming models.

Many industry initiatives are beginning to develop technologies to address just these problems that will increasingly impede progress toward tomorrow's more dynamic, distributed, mobile environment. Consider for example, Jini, Universal Plug-and-Play, and Service Location Protocol. As our contribution to these efforts, we decided to assess the functionality being provided by existing industry proposals, and to develop user performance and resource utilization metrics that can be used to assess performance of the various technologies. After analyzing some nine different proposed dynamic discovery technologies, we are developing a functional taxonomy that can be used to compare and contrast specific proposals. Further, we plan to deploy and measure performance of several of the more popular proposed technologies.

Achieving this paradigm shift will require researchers to begin thinking of information objects as active, mobile, and context-aware.

To provide an assessment of the ability of these technologies to work with tomorrow's SoCs we have developed a prototype, which we call the Aroma adapter. Each Aroma adapter consists of three PCMCIA cards, which together emulate the type of SoCs we expect to be available within five years. To date, we have used this prototype to convert a computer-controllable video projector into the type of portable, wireless information appliance we expect to be available in future years. We then implemented Jini on the Aroma adapter, converting the video projector into a Jini device that can be discovered, dynamically configured, and accessed across a pico-cellular wireless network. This prototype has given us our first hint as to the functionality and performance available from one of today's leading dynamic discovery technologies.

Rik: In your USENIX paper "AirJava: Networking for Smart Spaces" (Workshop on Embedded Systems, 1999), you mention the "Three Challenges for Smart-Spaces Researchers." Could you briefly outline these three challenges? (I particularly found the concept of location-relevant data in section 2.2 far-reaching.)

Kevin and Alden: The movement toward heavy use of embedded and mobile wireless SoC technology, supported by clusters of computers connected to both the wired and wireless infrastructure, will create new challenges that researchers and industry must address before users will benefit. Three challenges seem particularly important. First, mobile users carrying cell phones and personal digital assistants into spaces rich with information appliances will be able to discover a wide range of interface devices, such as large-screen displays, voice- and vision-recognition systems, and controllable cameras. Individual users and groups of users might like to exploit these distributed devices to construct ad hoc human-information interfaces to seamlessly exploit the best modes of interaction for the best purposes. We might call such interfaces poly-device, poly-modal (PDPM) interfaces. Some sort of distributed coordination bus will be required to compose such interfaces and to interact with information through them. Research will be busy working out the software issues and the human-interface issues presented by such environments.

A second challenge revolves around moving information for people. In today's model, either people carry their information with them, or they access that information remotely as they move from place to place. In either mode of operation, we must manage all our information for ourselves, even when much of the information we work with is context-dependent. For example, we typically attend meetings to conduct specific tasks. Before, after, and during these meetings we create information. Some of this information we retain personally, while other information is shared among the meeting attendees and others outside the group. Only a small fraction of this information is our own personal information.

Surely, as we move to the next meeting on the same subject we wish to have information from the last meeting available. At present, users must ensure that the necessary information is available at the right place and time. In the future, active information should be able to take on this responsibility. Imagine active information objects that can move, that can self-replicate, and that can communicate as a group. Such active information should be able to track the location, state, and trajectory of users, of object replicas, and of linked objects. In addition, active information objects should be able to plan the movement, replication, and transformation of information to serve the projected needs of its users, whether individuals or groups. Active information must also be able to implement consistency, access, and sharing policies among replicated and linked objects. Achieving this paradigm shift will require researchers to begin thinking of information objects as active, mobile, and context-aware.

A third challenge involves the transformation of information for presentation using knowledge about people, places, and devices. At present, network-based computing works because people carry in their minds a reasonably good model of cyberspace. We know that computers and printers can be found, and we know that information can be organized for storage on a disk. We know, but just barely, how to locate, download, configure, and execute various plug-ins to display information in specific formats or to convert information between formats. It appears that much of this knowledge is really rather routine, and that we could encode such knowledge into computers and networks, so that they could offload much of this routine work that we now require of the users of computers and networks. Using such knowledge, and the same heuristics that human users exploit, we could expect computers and networks to locate and compose devices and services in order to transform and display information in the best form for available output devices, personal preferences, and task effectiveness.

At least these three challenges must be overcome before users can be presented with effective, usable pervasive computing environments that leverage embedded and mobile, networked computer systems.

Rik: What about security? The idea of entering a Smart Space and being able to use a flat-screen display instead of my cell phone's puny LCD sounds great, but I can just imagine a confidential email flashing instead onto a large screen via the Aroma-enabled LCD projector. Are there any initiatives examining how to handle security?

Kevin and Alden: Good question. Security issues are currently unresolved, and along several dimensions. For example, given a set of embedded services, some of which should be made available to anyone who enters a space and others of which should be restricted to be used by particular individuals at particular times, how can access be controlled? This might be especially nettlesome because we can't expect every node to know about every conceivable user and to apply access restrictions on that basis. Probably work in the area of capability-based access-control mechanisms will help solve these issues. On the specific issue that you raised, users must be given mechanisms to set their own security policies (and other policies, for example about interruptibility). In my view, the composition of services and the routing of information to particular devices must be guided by policy mechanisms, some of which relate to user preferences, some of which relate to task, and some of which relate to context. These are all rich research areas that remain to be explored and solved.

The composition of services and the routing of information to particular devices must be guided by policy mechanisms, some of which relate to user preferences, some of which relate to task, and some of which relate to context. These are all rich research areas that remain to be explored and solved.

Rik: You mention a particular hardware implementation, called GUMPS. Could you describe what you are using in your prototypes, and perhaps provide URLs for sites selling something like this?

Kevin and Alden: We build two prototype adapters, both of which used components from various vendors. The components consist of four types: (1) Card PC, (2) PCMCIA wireless LAN interface card, (3) PCMCIA Flash RAM card, and (4) chassis and power supply. This approach to components is made possible by work on industry standards for PC cards for example: see <*http://www.pc-card.com/pccardstandard.htm*>. Card PCs are available from a number of vendors, such as Epson (<*http://www.epson-electronics.de/download/downcard.htm*>), Ampro (<*http://www.ampro.com/products/index.htm*>), and others. Flash RAM cards are available from a number of vendors, such as Kingston (<*http://www.amtron.com/price.htm#5*>), Synchrotech (<*http://www.synchrotech.com/products/ata-flash.html*>), and others. 802.11 wireless LAN cards available in two different technologies – frequency-hopping (<*http://www.proxim.com/*>) and direct sequence (<*http://wwwdb.lucent.com/bcs/*>) – spread spectrum, which are

I understand that a complete system with the same footprint as our assembled adapter can now be purchased off the shelf.

available from a number of vendors. The chassis and power supplies can also be acquired from multiple sources, typically the same sources as the PC cards. I understand from Alden that a complete system with the same footprint as our assembled adapter can now be purchased off the shelf.

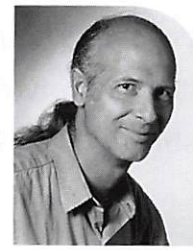
Rik: A final question. You had a great list of URLs at the end of your USENIX paper (although none for yourself). Do you have a Web page with that list of links (and perhaps others)?

Kevin and Alden: We had such a Web site; however, Sun Microsystems requested that we change the name of our adapter technology from the name used in the paper to another name. We chose Aroma. As a consequence, we took our Web site off the air, and we haven't gotten a chance to update the site yet.

musings on embedded systems

by Rik Farrow

Rik Farrow provides UNIX and Internet security consulting and training. He is the author of *UNIX System Security* and *System Administrator's Guide to System V*.



<rik@spirit.com>

I began this adventure into embedded systems when I started reading the proceedings for the 1999 Workshop on Embedded Systems (available from USENIX, of course). I thought I knew a lot about embedded systems, since that is where I began working as a nonstudent programmer, but boy, have things changed.

In this column, I will amuse you with my own experiences with embedded systems, and then venture into some of the new regions embraced by the embedded-systems movement. I think some of them will surprise you (they did me). The implications are staggering for those of you who manage networks, own homes, use PDAs, have a refrigerator, or drive cars.

Did I say refrigerator? The second paper in the proceedings postulates a future where embedded devices are truly ubiquitous. It describes a SmartCan that includes "a tiny computer, a small amount of memory, and a short-range radio transceiver." If that seems at all far-fetched, remember that Geode by National Semiconductor provides the first two (as well as a video controller and DSL) on a chip, and there are two existing transceivers-on-a-chip. The paper does omit the power supply, but a later paper describes schemes for broadcasting power . . .

In the SmartCan scenario, the embedded system remembers the make-up of the can, making it easy to recycle. It remembers the contents of the can and when those contents were stored (so that it can let us know when it is no longer safe to use). You can query your kitchen from the store and see if you have a can of stewed tomatoes or olives, and, if it is sitting in the refrigerator, how long it has been sitting there. Of course, I just can't wait until we have refrigerators that will be "intelligent" enough to start nagging us about the molding box of carry-out Thai sitting "on shelf 0, quadrant 4, and about to ooze over the vegetable tender." If the refrigerator is truly intelligent, it will try to discern our moods first (or, even better, hire someone to come in and clean it out without bothering us).

Welcome to the wonder of embedded systems. They already are everywhere. For every Pentium manufactured by Intel, there are somewhere between 20 and 100 embedded processors installed every year. You know that your VCR has one, as well as your microwave. Your new car or truck (er, SUV) likely has more than ten. Even a new blender has one, and of course your cell phone has a very zippy one. But this is nothing. In the future, the embedded systems will all talk to one another.

At least that is the plan. The big fight will be for the standard that unites all of the embedded systems. If you thought that the browser wars were something, just wait. Microsoft wants its software to be embedded in everything, and is making progress in that direction. (I can imagine the new excuse – I was late because my car crashed and had to be rebooted.) Sun's Java was always targeted, not at the Internet (that was an accident), but at embedded systems. I'll have more to say about Java later.

Ancient History

When I was a young man, I found computers fascinating. What I did not find so fascinating was the heavily controlled access to them. In 1978, someone left the specifications for the Zilog Z80 processor at my house while he wandered cross-country. I had abandoned computer work for something more physically stimulating (no desks for me), but I had a real awakening as I paged through the Z80 specs. This was a real processor on a single chip!

An average compile took about 15 minutes, giving me lots of time to read science fiction in between writing and testing.

After a bit of retooling at the University of Maryland, I quickly landed a job working for a company that was using the Z80 in an embedded system. The development system came from Zilog, and an average compile took about 15 minutes, giving me lots of time to read science fiction in between writing and testing.

The Z80 was an Intel 8080 clone with some additional instructions, for example, for block copies. This particular embedded system was designed for digitizing graphic data, for example, turning an aerial photograph into a set of labeled points. Here is how it worked.

The digitizer itself was a large table with a glass top. The underside of the glass had precisely aligned wires epoxied to it. The wires were attached to ground at one end, and to eight-bit shifters connected in series and a large power supply on the other end. A short pattern of ones gets shifted through the shifters, sending current down the wires, which generates a magnetic field. As the clock sent to the shifters is carefully regulated, this results in a magnetic field traveling across the digitizer's top, first for the *x* coordinate, then for the *y*.

To complete the setup, the operator must use a calculator-sized device to position a cursor over the point to digitize. As the magnetic field sweeps under the cursor, a coil detects the field, and the precise moment the field reaches a particular magnitude, the shifting stops and the number of clock ticks is recorded. By interpolation, the 1/4" grid of wires could measure to within three thousandths of an inch — theoretically. In practice, outside influences, including the magnetic interaction of the cursor itself, affected the readings.

The Z80's part in this had to do with collecting the number of clock ticks and converting it into *x* and *y* coordinates. This could be presented in several formats, as well as scaled or offset, and streamed out a serial port for data collection. A later effort included using Zilog Parallel Input/Output (PIO) chip to build a driver for a magnetic-tape unit (which actually worked).

In a later adventure, I consulted as a tech writer for Morrow Designs, one of the manufacturers killed by PC clones. I would come in, read the circuit diagrams and the code used by the embedded processor, and write the documentation. I liked this work because it was easy to do (while still stretching my mind a bit), and I did a much better job writing docs than I did coding large projects.

George Morrow often used microprocessors in disk controllers. He designed a floppy-disk controller that could read and write six different formats and used a Z80 to do almost everything. In another design (not by George this time), an early, ten-instruction RISC processor was used in a hard-disk controller with a channel-style control architecture.

With this sort of background, it is easy to see that I had a skewed view of what embedded systems look like today.

Appliances

A different view of embedded systems comes from network appliances. One of the first was Auspex, with a multiprocessor system supporting NFS and using either a Sun or an IBM workstation as a controller. Or Network Appliance itself, with its own operating system dedicated to file serving.

Then, there are the network devices, such as Ascend routers (BSDi), Juniper routers (FreeBSD), Cisco NetCache (BSDi), Big Ip from F5 (BSDi), the Linux router project,

and many more. The lure of rock-solid IP stacks and low cost (especially when compared to Microsoft), as well as high performance has made the BSDs and Linux the operating systems of choice when it comes to building network devices with some built-in smarts.

I have been asking around and hope that future issues of *;login:* will contain descriptions of some of these systems and how they were built. Also, there is a very active Linux community working on embedded-systems issues. You can find the Embedded Linux Consortium through <*http://www.linuxdevices.com/*>, as well as another embedded Linux portal at <*http://www.linux-embedded.com/*>. Both BSDi and Red Hat offer embedded-systems developer kits. The FreeBSD home page <*http://www.freebsd.org*> has a link to PicoBSD, a version suitable for embedded systems.

I think that examining UNIX versions used in these implementations is very useful, as it makes it clearer how you can build a true bastion host – a system stripped down to its bare essentials. For example, booting most *nices takes the boot loader, the kernel, /etc/init, and /bin/sh. Everything else is optional, although it's hard to get much done with *just* the shell. For a router, you might want to add gated and some management software.

An interesting addition in the Linux world is the BusyBox, a set of tools maintained by Erik Andersen (<*http://busybox.lineo.com/*>) and sponsored by Lineo, a company developing a version of Linux especially for embedded systems. BusyBox consists of a shell and other tools useful in a constrained environment (or what I used to think of as an emergency-repair diskette).

Other embedded systems include the much-maligned network computers. These diskless workstations are making somewhat of a resurgence. Personally, I would prefer to see nothing but diskless workstations on the desks sitting in office workers' cubicles. Take away the disk, and you take away most of the maintenance and administration problems, and do wonders for security. Of course, you move that to a central server, which, while much easier to administer, creates an attractive single point of failure. Also, the workers will be really pissed off when you take their games away from them.

Then, there is the really basic embedded system, such as WebTV. I got to set one up the other day (don't laugh). I had just gotten a massage, and the masseuse mentioned that she had just bought a WebTV but was worried that she couldn't set it up. Her problem wasn't the setup, but the array of cables needed to put it together. The masseuse had already set up her new VCR, connected the cable input, and got the TV working, so I wondered what was so difficult about the WebTV.

For me, it was nothing, just standard cables, power supplies, the phone cable, and a printer. Once everything was connected, we turned on the power and the system first dialed an 800 number, found the nearest point-of-presence, hung up and redialed a local number, and started stepping through the sign-up instructions. I left at that point, but called back later to see if she had been able to finish the sign-up and send email. Boy, was she excited. The system was really working for her, and even the printer worked (an addition to the system, and it just worked).

The Future

You probably see the problem. It wasn't the software, which guided my friend through unerringly. (I was impressed.) It was that mass of unfamiliar cables that had her stumped. Really, everything was simple, as the audio/video cables were color-coded, and the printer cable could only fit one way. Trivial, for an engineer.

There is a very active Linux community working on embedded-systems issues.

I believe Java, and the set of specifications for embedded systems, Jini, is very real.

But this business about connecting things together and configuring them so they will work together is exactly the thing that future embedded-systems designers want to avoid. The best-known example of this is what are called Smart Spaces. A Smart Space is a wireless island where available services and devices discover one another, do whatever self-configuration is necessary, and adjust to changes that occur in the environment, all without the intervention of a system administrator (or masseuse or her client).

This part really fascinates me. Imagine walking into a conference room, having your laptop discover the LCD projector, the printer, the local nameserver, and the gateway to the Internet, all without you doing anything at all. You make your presentation using the LCD projector, perhaps print a couple of pages, while having access to sites on the Internet without doing any configuration at all. I know that DHCP takes us part of the way there (automatic IP address assignment, along with the nameserver and gateway address), but what about talking to the projector and the printer?

Back in 1994, I had the opportunity to interview Bill Joy in Aspen. I was playing at being a print journalist, as the real journalist assigned to the interview hadn't been able to understand what Joy was trying to tell him. Joy actually can be crystal clear (see his April *Wired* article entitled "Why the Future Doesn't Need Us" <*http://www.wired.com/wired/archive/8.04/joy.html*>). Joy repeats a chilling view of the future of humanity when computers become smarter than humans – from the Unabomber's manifesto. Joy goes on to describe the dangers inherent in gene tailoring as well as nano-technology. A very thoughtful piece, and, given the current level of sanity in the world, it left me feeling rather gloomy about the not-too-distant future.

But back to embedded systems. In true paparazzi style, I had cornered Joy at a think-tank meeting (the Aspen Institute) and asked him if he would meet with me to complete the interview. We did get to meet in an outdoor cafe, where Joy tried to explain to me (without giving away Sun secrets) his view of the future – Java.

After being bored by my obvious questions about the beginnings of BSD, Joy got very intense, first when talking about future design of SPARC architectures. It was nice to be talking to someone who actually was creating designs for five or more years into the future. But the other thing he started talking about perplexed me at the time. Joy pulled out his cell phone. "There is no reason why I shouldn't be able to get this cell phone to work with a printer by just setting it next to it," said Joy. At the time, I thought, big deal, he's talking about infra-red networking. Of course, I was wrong. Joy was talking about Java, something it took me years to realize.

Joy's vision of the future had Sun software (and perhaps hardware) sitting in the middle of it. This vision has been somewhat derailed over time, but I believe Java, and the set of specifications for embedded systems, Jini, is very real. Again, you can read *Wired* (<*http://www.wired.com/wired/archive/6.08/jini.html*>) or visit Sun's own site (<*http://sun.com/jini/*>).

Before getting into Jini, I'd like to back off a bit, and talk about the requirements for Smart Spaces. Ideally, in a Smart Space world, there are no cables. Everything has its own networking, and just bringing devices into proximity is sufficient for configuring the devices. Thus, if the WebTV I installed had been Smart Space–ready, all it would have taken to get it working would have been to turn it on.

I must pause at this point, because the thought of all this wireless self-configuration going on gives the security person in me the heebee geebees. Given the software indus-

try's track record for designing secure software (zip), this whole idea scares me silly. If I don't have to connect the keyboard, how do I know which server I am typing to (or how many, because I am truly paranoid)? Cables provide a warm-fuzziness, as I know that the keyboard is physically connected to that workstation, and that if I have protected the X-server properly, I don't have to worry about my keypresses winding up somewhere else.

Perhaps I am being a bit extreme when I write that no secure software has ever been written. There was hello.c, and lots of work has been done to make sendmail secure, and to get the bugs out of SSH. The other factor that frightens me is the market's willingness to accept and use insecure solutions. Today the firewalls of choice specialize in performance, not security. E-commerce sites will initially dispense with having any firewall at all, because getting up and running is all-important prior to the IPO. And, of course, firewalls are not all there is to security – they just cover one door into the organization. Sigh.

Back to Smart Spaces. Devices in a Smart Space are supposed to identify themselves to the other devices and to be able to find what they need. For example, imagine you are sitting in the terminal room at a USENIX conference, and you want to print something. If it were a Smart Space, you would just print, trusting your operating system or application to locate a nearby printer, load the correct device driver and filters for it, and queue up your print job. And if, the next time you go to print, that printer is no longer available (perhaps it is out of paper), the Smart Space specs will not let you down, but will seamlessly find another printer and go through the self-setup again, without intervention.

Suppose the entire conference center is a Smart Space. Forget about the terminal room, as wherever you go, you will be connected to a network within the conference center, which will provide you with a nameserver and gateway to the Internet. As you roam, your IP address will change, but your connections will persist. You may have read about these issues in USENIX papers about mobile computing, but they are part of today's embedded-systems world as well.

There is also the notion of application dependence. This may not be a problem if everyone uses Microsoft applications and keeps them updated to the most recent version. Gag. More realistically, suppose the person you are meeting with wants to transfer his or her new opus, but it is written in Word, and all you have is a PDA. If the Smart Space is *really smart*, it will transparently handle converting whatever application-specific format the document is in into something your system can handle.

That seems to be asking a lot from Smart Spaces (perhaps they should be called "Genius Spaces" instead). Kevin Mills, in his paper, goes further when he talks about having data that is tied to a particular context, for example, a committee meeting. Whenever and wherever that committee meets, its minutes and other appropriate information remain accessible. Now that is really out there. But so were 10-gigabyte hard drives for under $200.

Jini

Some of these problems have already been successfully attacked through Java Jini, a set of specifications and interfaces for discovery and sharing of networked services. O'Reilly & Associates had kindly sent me a copy of *Jini in a Nutshell*, by Scott Oaks and Henry Wong. I had asked if the authors wanted to write this part, and got the book instead. So here goes. We are all most certainly aware that any Java Virtual Machine

If the Smart Space is really smart, it will transparently handle converting whatever application-specific format the document is in into something your system can handle.

The [Jini] specs define interfaces that must be supported for Jini-enabled services. The goal behind these interfaces is that services will be able to interact in a dynamic fashion without human interaction. No setup, administration, or guidance.

(JVM) should be able to run any compliant Java code. This concept is one of the founding ideas behind Java – write once and run anywhere. But it will take more than that to make Smart Spaces work.

And this is where the Jini specifications come in. The specs define interfaces that must be supported for Jini-enabled services. The goal behind these interfaces is that services will be able to interact in a dynamic fashion without human interaction. No setup, administration, or guidance. These features can be summed up as follows (quoting page 4 of the book):

- The Jini environment requires no user intervention when services are brought online or offline (other than starting the service, e.g., turning on the Jini-enabled device or starting the Jini software service).
- The Jini community is self-healing: it can adapt when services (and consumers of services) come and go.
- Consumers of Jini services do not need prior knowledge of the services implementation. Instead, the consumer loads the service implementation dynamically, with no configuration or user-intervention required.

The book goes on to explain how to download the free developer's kit, which requires version 2 of the Java Developer's Kit (JDK 2). Sun provides a registry service (reggie), a transaction service (mahalo), temporary storage for objects in JavaSpaces (outrigger), a lookup service (fiddler), an event mailbox (mercury), and a lease-renewal service (norm). When I encountered these names, I wondered if the Jini team had holed up in Hawaii instead of Mountain View or Aspen.

The authors go on to point out that Jini does have support for security, something not available with Microsoft CE implementations. (Sorry, digital signatures, as found in DCOM, are not sufficient protection.) Jini security mechanisms are identical to JDK 2 and permit access control to all system resources based on the classpath, codebase, and/or digital signature. Chapter 12 of the book goes into a little detail about security. For more details, you must read the book *Java Security*, written by Scott Oaks and published by O'Reilly in 1998. Not all of the over 400 pages have to do with access control; there are also chapters on encryption, message digest, and other topics. What pleased me is that Jini does have support for fine-grained security.

The authors go on to describe installing, setting up, and starting host-based Jini services. Once you get all of this going, you can begin writing your own Jini services, using the examples found in the book. I am no longer proficient in Java, but the book appears to be thorough. If you have had good or bad experiences using this book, send me some email, and I will share it.

Aroma

A related project mentioned in the proceedings was AirJava, renamed "Aroma" after Sun complained. Kevin Mills of NIST plans to create a working pico-cellular network-based test platform in the near future. The goal would be to create a functioning test-bed that included a processor, the network interface, some memory, and perhaps a PC card interface for those who wanted to get some practical experience with Jini. See my interview with Mills and Dima in this issue of *;login:*.

Looking again at the proceedings, there are papers about massively distributed systems (just imagine your car's ignition system processor participating in the search for extra-terrestrial intelligence [SETI] while you are stuck in traffic), learning algorithms (intelligent embedded systems with pattern recognition), and virtual user interfaces (a

method for creating a user interface for anything your PDA finds in the pico-cellular network neighborhood). I particularly liked this last one (Kangas and Röning <*macconen@ee.oulu.fi*> and <*jjr@ee.oulu.fi*>, as they explicitly mention the "limited functionality" of VCR and microwave-oven displays. They also point out the limitations in input as well, and include cell phones in their lists of devices.

There are other systems I have not mentioned here. QNX, a realtime OS, can be used in embedded systems, and Inferno will be used in telco equipment (unless Lucent goes out of business, which I really doubt). Microsoft is not going to rest while this goes on. (It does own WebTV, and I will confess that the user interface passed the masseuse test.)

I have always wanted to live in an intelligent house and drive an intelligent car, and am somewhat disappointed at the slow pace of change. Using X10 controllers to turn lights off and on is not very exciting to me, but having access to information wherever there is a telephone in my house is. Why should I have to find the PDA or the Rolodex just to call someone, or to add toilet paper to the shopping list? Why doesn't my car have a nice GPS in it (because it would be stolen?), as well as a method for self-diagnosis that doesn't involve coded beeps or display lights flashing?

We are living on the cusp of a new age. I decided to write about this because it excites me, but also because I believe it will affect the USENIX community before it affects the general public. That is because we manage a lot of the computing and networking infrastructure in the world today, and embedded systems may well add to our load.

And finally, I want embedded systems to be based on open standards with good designs. These standards must include support for security and for extensibility, and not stifle innovation through complexity or license fees. Examining history shows us that the standards are most commonly based upon whatever is widely used (and you wonder why I avoid using Microsoft products?). The process for creating the standards for the future of embedded systems is happening now, and it is important that we be at least aware of it.

Oops, gotta go, my refrigerator just opened a window on my screen telling me that the ale I put in it has reached 10 degrees C. If you do want to hear more about embedded systems, let me know.

I want embedded systems to be based on open standards with good designs. These standards must include support for security and for extensibility, and not stifle innovation through complexity or license fees.

teaching operating systems with source code UNIX

by Bob Gray

Bob Gray is co-founder of Boulder Labs, a software consulting company. Designing architectures for performance has been his focus ever since he built an image processor system on UNIX in the late 1970s. He has a Ph.D. in computer science from the University of Colorado.

<*bob@cs.colorado.edu*>

Thanks to Charles B Morrey, Art Messal, Mike Durian, Steve Gaede, and the students of OS3753.

It's early April and the end of spring break at the University of Colorado. I have a couple more days before the next lecture to finish grading the programming project that illustrates a file's on-disk layout. I also must work through the next project – implementing "undelete" in the kernel before assigning it to 95 college juniors.

Back in December, the computer science chair asked me to teach the undergraduate operating-systems course. I asked what resources were available to me and was offered a set of notes from the previous semester. That course's homework was based on binary-only Windows NT. I inspected the material and found the assignments were just "userland" programming projects.

I chose to teach the course using a source-code, production operating system. This article discusses the advantages and disadvantages of my decision. I'll outline a couple of the programming projects, with the lessons learned, and I'll pass along some of the students' comments. The course assignments and solutions are posted at *<http://boulderlabs.com/os3753>*.

Introduction

A long time ago, my undergraduate operating-systems class required that we cross-compile a small, standalone system and upload it to a PDP-11 minicomputer. We could do some limited debugging at the console if the program didn't crash. The development environment was poor; it was painful and time-consuming to get things working, but the experience was an overall confidence builder. I feel there is a huge advantage for a student to control the operations of a computer directly.

Another approach for teaching operating systems is to provide a controlled runtime and development environment using a simulator. Several universities teach operating-system concepts using the Nachos simulator (*<http://www.cs.washington.edu/homes/tom/nachos>*). The advantage is that the instructor can easily control much of the environment for assignments, and the students don't waste time with crashes, kernel builds, and rebooting. These kinds of systems can be very simplistic and lack realism.

As a private pilot, I know that aviation simulation goes only so far. You need to spend some time in the sky, in the air-traffic-control system, in the weather, and with the attendant dangers, to absorb and appreciate the training fully. A two-hour actual flight lesson is often fatiguing and draining; but the same amount of time in a simulator is more like a classroom experience. Similarly, students sense the difference between working in a safe simulator environment and working on a real kernel. Lessons with the latter seem more dramatic.

Source-Code Operating-System Course

The theory and concepts of operating systems at an undergraduate level are not difficult or time-consuming. We quickly covered them with the excellent textbook by Silberschatz and Galvin, *Operating System Concepts* (Wiley, 1999). The implementation is the difficult part, and I claim that until someone actually digs into the code, they haven't fully learned the important stuff.

I wanted my students to gain an appreciation of how "it really works"; therefore, a source-code operating system was required. Most students have an Intel-compatible

personal computer, so any of the Linux or BSD distribution would have been fine. I standardized my class on FreeBSD, thanks to a donation of 100 CD-ROM sets from Walnut Creek CDROM (<*http://www.cdrom.com*>).

Some students groaned about loading another operating system onto their machines, but firmly restating my plan and handing each of them a new four-disk package quickly ended the discussion. I wouldn't want to introduce more complexity in the course by encouraging variations. The university supplied a few old Pentiums for several students who didn't have equipment. For them, we installed big disks so that a couple of students could share a machine and still each have their own, unique disk partition.

The first operating-system project was to load the system and configure a custom kernel for their hardware. For most, the project was fairly easy, but given all of the variations of cheap PC hardware, a number of problems arose. A high percentage of the class needed to split their Win9x partition to make room for UNIX. Some older BIOSes have the constraint that you cannot boot an operating system if it is beyond cylinder 1023 or beyond 8GB. Also, there were the usual difficulties with various graphic chips and certain Plug'n'Play hardware.

After two weeks, almost all students had completed the first assignment and were running solid systems. My role in assisting in their success was to provide written resources and some of my time. I offered to help overcome loading problems if the students would carry their PCs to an on-campus lab. (They didn't have to carry their monitors.) I enlisted the help of several friends and system administrators to accelerate the process of handling all of the weird or problematic machines. Bringing up 95 machines was a lot of work for me and some of the students.

For each programming assignment, I've asked the students to submit a report containing these sections: Introduction, Methodology, Analysis, and Summary. The idea is to put on paper the essence of the project – their target audience is former operating-system students. They include enough information that, armed with the assignment specification sheet, some other competent programmer could obtain the same results.

I encourage students to speculate on what is going on in the system and to substantiate their hypotheses with measurements. If they don't have the time or resources to investigate certain phenomena, I ask them to at least suggest additional experiments that could help describe the system.

Even though I have a teaching assistant and a grader for my course, I also read the students' reports. I'm delighted when some of them make subtle observations or raise interesting questions. Sometimes I find logical thinking and reasoning but incorrect conclusions. For these situations, I point out the problem and try to get them back on track. Here's where the ratio of students to instructor is crucial – there must be enough time to understand the difficulties and to handle them.

More than once, I had handfuls of students with the same inappropriate mental model. Because they have gone to the trouble of explaining the model on paper, it is both easy and dramatic to cover the misconceptions in a lecture. For example, some students observed that the filesystem block addresses were increasing by 8 for sequential pieces of a file. They hypothesized that the file system block layout was "interleaved" to allow time for disk-controller set-up between I/O requests without losing a revolution. I pointed out that "interleaving" is no longer necessary with today's disk controllers, which are always reading into their hardware caches. So even if a request for a sector occurs after it passes under the reading head, the data will be immediately transferred

For each programming assignment, I've asked the students to submit a report containing these sections: Introduction, Methodology, Analysis, and Summary.

Most students seemed to enjoy the challenge of working in the kernel, even though it turned out to be time-consuming.

to the computer. The address jumps actually represented contiguous blocks – the FFS allocates blocks in 8KB pieces, addressable in 1KB units.

Because production operating systems have been highly optimized, it is a challenge to create student projects that are useful, interesting, nontrivial, but not too hard. Much of the kernel code base is superfluous to the important course concepts and can overwhelm individuals. I'd say that this is the main disadvantage of my approach. However, I found that by working a solution beforehand, I was able to guide students along a productive path and save them from numerous traps. As necessary, I provided code fragments or other hints to focus their efforts. Most students seemed to enjoy the challenge of working in the kernel, even though it turned out to be time-consuming.

Kernel Instrumentation Project

One of the early projects involved inserting instrumentation into the kernel, running a synthetic load, and analyzing the results. We measured the time to create a new process (fork) and observed what factors influenced the event. Students learned about the high-resolution timer available on PCs and gained some knowledge about statistics. In the kernel, they learned about the cost of copying a process's environment and how to add instrumentation wherever needed. Here is a typical student summary (given with permission):

> From this exercise I have learned quite a bit about the creation of a new process and how it is added to the runnable queue. It appears that the creation process takes generally the same amount of time for each process. . . . When doing this assignment, I wondered if the forking process takes longer when more CPU intense programs are running. If I had more time, I would investigate the question by creating tight algorithms that hog the CPU. If I had more time, I would also like to check the time for each individual subroutine calls of the fork1 function to determine what takes the most time. By doing so I could also determine which ones remain completely constant and which ones vary slightly. This would help pinpoint the important parts in the fork process.

File System Project

The latest project required that the students print the filesystem block address for selected files. They had to learn a lot in a short amount of time. Their main resources were include files and man pages. A gratifying part of teaching is reading student conclusions such as the following:

> The FreeBSD file system is a complicated beast when you first look at the source code. However, as you page though the seemingly endless code, man pages, and header files, the logical structure emerges. Along the way, I encountered several problems that I still do not understand. . . . This assignment was a great learning experience, however, it was, by far, the most time consuming task I've encountered in this class.
>
> In general, the Berkeley Fast File System seems to be a great performer, lumping large portions of files together contiguously in order to try to minimize the number of I/O operations required to read them. Also, only a very small percentage of disk space is wasted in indexing these block addresses. Random (direct) access to disk blocks and sectors is very efficient, requiring only 3 I/O operations at most to get to any given disk block. There is less external file fragmentation since portions of files can reside in any size hole, and very little internal fragmentation since each filesystem block can be broken down into fragments which are each individually addressable. This is leaps and bounds better than pure linked allocation or pure contiguous allocation. UNIX

provides an efficient file system that can be navigated to explore how and where files are stored on disk. I followed the steps listed in the assignment closely and found it hard to understand everything I was seeing, but having to explore and interpret the file system on my own increased my confidence to explore a complex OS. This was a pretty cool assignment, I don't think that I've ever read as many man pages and header files. It is interesting that with operator privilege you can get in otherwise "permission-protected" files. In summary this was a good exercise to enhance one's understanding of how the raw disk management of a file takes place. It is so easy [to forget that] behind the glitz and glamour of Window's and Linux's flashy GUIs and program managers all of those files are actually just strips of magnetically encoded information on a series of spinning disks. It would be interesting to conduct tests to see how the results of our program would differ as a disk's fragmentation level increased, as contrasted with a freshly defragged disk.

Virtual-Memory Project

For one project, I had the students add kernel code to capture information about page faults. I then supplied three "mystery" binary programs, and I asked them to figure out what was going on with respect to the virtual-memory system. Here are a couple of pieces of their summaries:

> I found this assignment to be eye-opening. The virtual memory concepts were always abstract in my mind, but now that I've seen the graphic representation of its behavior, it makes much more sense. In fact, now I wonder why I had such a complicated mental image before.

> In doing this lab, I learned quite a bit about the virtual memory system and how to analyze programs. I found this assignment very interesting and helpful to my diagnosis skills. By looking at the three mystery programs, I could see how the operating system deals with the access of variables and it was much more complicated than I imagined. From program 1, I was able to observe sequential accesses coming from possibly three different arrays or possibly a recursive call. From program 2, I was able to observe an array that was quite high in the stack and must have had other values that were initialized before it. Finally in the third program I was able to observe random access followed by a backwards sequential access of a different array. If I had more time, I would like to find more about the gaps in my graphs which were probably due to interrupts and disk accesses. I would also like to find out what each program accessed near the 16000 and near the 79000 range. From what I can tell it must be some type of overhead that the operating system has to go through with a process.

> In summary I can say that this was a very successful experiment. By placing instrumentation into the kernel we were able to confirm some of what we know about how the contents of a program are laid out in memory. The text and global data segments are in the lower part of memory with the heap growing upward from there and the stack sits in the highest memory and grows downward. It is important to note, however that virtual address space is different [from] the physical address space. The pages that we were looking at could be anywhere in physical memory and do not have to be laid out in such a nice order as implied by our graphs. It was most interesting to try and create a program that attempted to simulate what some of the program had done. It was this experimentation that gave me the most insight into the differences in a global variable and a local variable, the heap and the stack, and where the code segment lies.

I supplied three "mystery" binary programs, and I asked [the students] to figure out what was going on with respect to the virtual-memory system.

Using source code UNIX has been successful for both teaching operating-system concepts and for building confidence for working in a large code base . . . enrollment increased 15%.

This assignment was the best learning experience yet. I read a lot about VM, and was reminded of the fact that "main" is just another function with its own stack for its own "local" variables. I tend to think of main's variables as global, but they do reside on the stack – as the graph of my own thrasher showed. It would seem that only those variables declared outside of main appear low in memory, and all others reside on the stack.

Summary

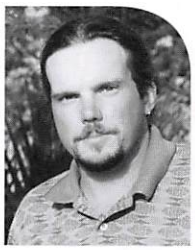
I feel that my approach of using source code UNIX has been successful for both teaching operating-system concepts and for building confidence for working in a large code base. Before the class started, I had expected enrollment to drop once the students learned how much work was expected. Instead, word got around that this semester's course was going to be substantially different from the previous systems-programming course, and enrollment increased 15%. Any course that requires programming suggests a lot of work for both the instructor and the students. I've gained a lot from this experience and I suspect the same is true for many of my students. Next time there is the opportunity, I'll again employ source code UNIX for teaching operating systems.

effective perl programming

Manual SQL (It Rhymes)

by Joseph N. Hall

Joseph N. Hall is the author of *Effective Perl Programming* (Addison-Wesley, 1998). He teaches Perl classes, consults, and plays a lot of golf in his spare time.



<joseph@5sigma.com>

You Too Can Write an SQL Client

Lately I've found myself spending an increasing amount of time working with Perl and SQL databases, and sometimes with more than one type of server at a time. One minor aggravation in dealing with different kinds of servers at the same time is that their command-line clients work differently.

For example, MySQL's command-line client, mysql, has the GNU readline library built into it, which means that you can use the up and down arrows (or Control-N, Control-P) to access the command-line history, and various emacs-like commands to edit the current line. Oracle's SQL*PLUS client, on the other hand, has a lot of nifty features, but no readline library. Ugh.

Well, I guess if my SQL client(s) don't suit me, I should consider writing my own. And that's exactly what I've done for this article. In years past I probably wouldn't have considered writing my own SQL client, and certainly not as an afternoon "quickie," but as you'll see below, nowadays with Perl it's just a matter of slapping together some modules.

Starting Up

First, make sure you have the DBI, Term::ReadLine, and Term::ReadLine::Gnu modules installed, as well as the DBD module(s) for your favorite server(s).

Our shiny new *server-independent* SQL client will be called perlsql. Let's start it off like this:

```
#!/usr/local/bin/perl -w
use strict;
use DBI;
use Term::ReadLine;
use File::MkTemp;
```

The use DBI directive gives me the DBI module, and use Term::ReadLine gives me an interface to GNU readline-like functionality. File::MkTemp will also come in handy in a bit.

From the UNIX command line, we'll invoke perlsql something like this:

```
persql 'DBI:Oracle:host=localhost;sid=main' scott/tiger
```

The first argument is a DBI DSN string. It will, of course, vary (considerably) depending on what server you're connecting to, how it's set up, and what environment you are executing in. The second argument is an Oracle-style username/password identifier. Here's how we process the command line:

```
my $dsn = shift;
die "usage: perlsql dsn [user/password]" unless $dsn;
my ($user, $passwd) = split(/\/, (shift || ''));
```

The shift operator works on @ARGV by default if you don't specify an argument. The username and password default to empty string if none are specified. Now we're ready to connect to the database:

First, we check to see if the line is an SQL command. The . . . list of SQL commands . . . is obviously server-dependent, but simple enough to come up with.

```
$dbh = DBI->connect($dsn, $user, $passwd,
    { PrintError => 0, RaiseError => 1, AutoCommit => 1 }
) or die "can't connect:\n$@";
```

This connects us to the database and sets some connect attributes. We turn the PrintError attribute off so that error messages aren't automatically printed. Turning RaiseError on causes DBI to generate an exception when an error is encountered. Turning AutoCommit on automatically commits every statement executed by DBI. Next, let's initialize Term::ReadLine:

```
my $term = new Term::ReadLine 'PerlSQ';
my $OUT = $term->OUT || *STDOUT{IO};
```

We're now ready to write some command-processing code.

The Readline Loop

I'll go ahead and show you the entire command-processing loop, and then explain it a piece at a time.

```
while (defined($_ = $term->readline("$line> ")) ) {
    my $cmd = $_;
    $cmd =~ s/^\s+|\s+$//g;        # lop off whitespace
    next unless $cmd;              # skip blank lines
    my $first = (split(/\s+/, $cmd))[0];
    if ($is_sql_cmd{lc $first}) {
        do_sql($cmd);
    } else {
        if (lc($cmd) eq 'quit') {
            $term->remove_history($term->where_history);
            last;
        } elsif ($cmd =~ /^!/) {
            system substr $cmd, 1;
        } else {
            eval qq(
                package perlsql; no strict; \$save = select(STDOUT);
                \$res = do {$cmd}; select \$save; \$res
            );
            print $@ if $@;
            print "\n";
        }
    }
    $line++;
}
```

The command-processing line is, overall, a while loop that reads a line at a time from our ReadLine terminal. If the user types the end-of-file character, $term->readline returns undef and drops us out of the loop. Inside the loop we first strip leading/trailing white space from the command line and make sure it's not blank. If it's not, we process the line in one of several possible ways.

First, we check to see if the line is an SQL command. The hash %is_sql_cmd contains a list of SQL commands. This is obviously server-dependent, but simple enough to come up with. I define it like this:

```
my %is_sql_cmd = map { $_ => 1 } qw(
    alter analyze associate audit call comment commit
    create delete dissociate drop explain grant insert
```

```
    lock noaudit rename revoke rollback savepoint select
    set truncate update
);
```

If it is an SQL command, I pass it to my do_sql subroutine, which I'll explain below. The next possibility is that the user has typed quit. In that case, I delete the current line (containing quit) from the readline history, and exit the loop. Another possibility is a line beginning with an exclamation mark. Those lines get sent to a shell with Perl's system operator.

If the line doesn't fit one of those descriptions, it's treated as a Perl command. If you supply the eval operator a string, Perl takes the string and executes it as Perl code in the current context. Note that I'm using the generalized qq form of double quote – it just looks better to me than ordinary double quotes if I'm quoting several lines.

I don't want the Perl code executed in my current package (otherwise, commands typed in by the user might inadvertently mess up the running perlsql program!), so I change to a different package in the eval – perlsql in this case. I turn off strict and make sure the default filehandle is set to STDOUT, then execute the command line in a do block. Then I restore the default filehandle and return the result of executing the command. (Note that this code doesn't actually do anything with the result, $res, but that's a feature that could be added.) If the eval produced an error, the message will be in the $@ variable, so I check that and print it if necessary.

At the bottom of the loop, I increment the line number counter.

Processing SQL

The do_sql subroutine takes a single SQL command as its argument, double-quote interpolates it, executes it, and then displays the result.

The do_sql subroutine takes a single SQL command as its argument, double-quote interpolates it, executes it, and then displays the result.

```
sub do_sql {
    my $sql = shift;
    $sql = eval "package perlsql; no strict; qq\0$sql\0";
    print($@), return if $@;
    print "=> $sql\n";
    my $sth;
    eval {
        $sth = $dbh->prepare($sql);
        my $rv = $sth->execute;
        if ($sql =~ /^\bselect\b/) {
            my ($prec, $names) = @{$sth}{qw(PRECISION NAME_lc)};
            display_result $prec, $names, $sth->fetchall_arrayref;
            print "\n";
        } else {
            print defined($rv) ? ($rv + 0, " rows affected.\n\n") : "ok\n\n";
        }
    };
    print $@ if $@;
}
```

Double-quote interpolating SQL command lines is useful because it lets us use Perl variables inside our commands – something like:

```
select count(*) from club where state = '$state'
```

To double-quote interpolate $sql, I eval it in the perlsql package. Note that the argument to eval is a double-quoted string, and that within that I have another double-

To enter rather long commands . . . just bind the Control-V key to a subroutine that lets you edit the current line in your favorite editor.

quoted string. I use a NUL (\0) as the delimiter for the embedded double-quoted string. After that, I echo the interpolated command to the terminal and then prepare and execute the command.

I display the results of select statements with the display_result subroutine:

```
sub display_result {
    my ($prec, $name, $ary) = @_;
    my $row_format = join(' ', map "%-${_}s", @$prec) . "\n";
    printf $row_format, @$name;
    print join(' ', map { '=' x $_ } @$prec), "\n";
    printf $row_format, @$_ for @$ary;
}
```

The arguments to display_result are array references containing the "precision" of the result columns (how many characters are required to print the contents), the names of the result columns, and the results themselves. The results are a two-dimensional "array of arrays." I use the precision to create a suitable printf format (just printing the data as strings) and then print each row of the result. The whole thing winds up looking like this:

```
112> select distinct postal_code from club where name like 'Augusta%'
=> select distinct postal_code from club where name like 'Augusta%'
postal_code
===============
30904
04351
67010
46701
```

For non-select statements, I print the number of rows affected by the command (if available).

More Powerful Command-line Editing

One of the annoying things about using SQL command-line clients is that you often need to enter rather long commands. Perhaps you'd like to be able to edit them using a separate editor? No problem! We'll just bind the Control-V key to a subroutine that lets you edit the current line in your favorite editor:

```
$term->add_defun('visual', sub {
    my $fn = mktemp("perlsql$$.XXXXXXXX", "/tmp");
    $fn = "/tmp/$fn";
    open F, ">$fn" or die "can't open $fn: $!";
    print F $term->copy_text;
    close F;
    system +($ENV{EDITOR} || 'vi'), $fn;
    if (-r $fn) {
        local $/;
        open F, $fn;
        my $text = <F>;
        close F;
        $text =~ s/[\r\n]+$//;              # no trailing newline
        $term->begin_undo_group;
        $term->delete_text;
        $term->Attribs->{point} = 0;
        $term->insert_text($text);
        $term->Attribs->{point} = length $term->Attribs->{line_buffer};
```

```
            $term->end_undo_group;
            unlink $fn;
        }
        $term->forced_update_display;
} );
$term->bind_key(ord("\cv"), 'visual');
```

The Term::ReadLine::Gnu method add_defun registers a subroutine with the readline library. In this case, I've defined it as an anonymous subroutine (with the sub {} operator). I use the mktemp subroutine from File::MkTemp to create a temporary filename, then create a temporary file with that name, write the contents of the current command line into it (from copy_text), and fire up an editor on that file with system. If the editor leaves a readable file, I read the contents back in as a single blob of text (clearing the $/ special variable makes Perl ignore line endings when reading from the file) and use that to set the current command. I found that it was necessary to manipulate the insertion point with Attribs->{point} manually to avoid some weird problems. A call to forced_update_display after everything's done forces the readline library to update the display.

The bind_key method binds the subroutine that I've registered with the name visual to the Control-V key.

Cleaning Up

An END block handles disconnect from the database and cleanup of any temporary files that might have been left behind:

```
END {
    print "\n";
    $dbh->disconnect if $dbh;
    unlink </tmp/perlsql$$.*>;
}
```

Features Gone Begging

I've written a slightly longer version of this program that has a few more frills. It saves the history to a file and restores it on startup, and also reads in a ~/.perlsqlrc file written in Perl on startup. You can see it in its entirety at *<http://www.perlfaq.com/examples>*.

This short program (the version on my Web site above is only 140 lines long as of this writing) is, I think, an excellent demonstration of how you can quickly create surprisingly powerful and useful things in Perl. It took me only a few hours to write perlsql. Yet, even after that small amount of work, it's a useful database-independent SQL client, and one that knows Perl in addition to everything else!

The idea behind perlsql isn't a new one – there have been previous attempts at writing DBI/ReadLine clients. The notion hit me all on my own but it was of course not original. The first such well-known DBI-based client was Andreas Koenig's pmsql. A later program was dbimon. dbimon is apparently out of date, but I have also seen a few other more recent Perl-based SQL clients.

You can quickly create surprisingly powerful and useful things in Perl.

firewalls at home

by John Sinteur

John Sinteur is the maintainer of the free NetBSD/i386 Firewall project at <*http://www.dubbele.com/*>. John's current day-time activities include being contracted out by his employer, Aedius IT diensten, as a lead developer for e-commerce applications.

<*john@dubbele.com*>

The Problem, Or, Times Are A-Changing

Firewalls have traditionally been pretty scary things – devices that sat in protected areas in the computer room, maintained through High Wizardry, protecting the Company Network from All Evils. However, as we have seen with the DDoS attacks earlier this year, computer security is not something that should be limited to large companies with big, fat pipes coming off the Net. Neither is computer security something that is being done solely to protect the company network from what's happening outside. Not a great many people are aware that nowadays the Internet as a whole can be in trouble when a single system, yours, is compromised. With tools like Tribal Floodnet, TFN2K, and Stacheldraht, your system is not the one that is hurt the most when it is compromised. Your system could be used to participate in the next round of DDoS attacks that hits the news, or it could be used to send out spam.

A big change is happening right now: it's no longer just companies or universities that have fast enough permanent connections to be of interest to the average script kiddie out there. With cable modems and ADSL lines being installed in huge numbers, there's lots of bandwidth connected to unprotected systems. People at home never had to think about this kind of thing before. They are not aware that they are the next big target for script kiddies. The number of systems that can be compromised easily is exploding. And can the home user really be blamed? I don't think so. What's special about an average family with kids in high school owning more than one computer in the home, and connecting them together in a small network to share a hard disk or a printer? Nothing really, but unless they take specific action to prevent it, those drive shares will be accessed from across the planet. Any sysadmin who has installed snort[1] on his firewall can testify that Windows shares are the most often scanned for "vulnerability."

Security at Home

Companies should not assume that as long as their computer network has a good firewall and security policy, employees' home computers are a nonissue. In the good old days when people dialed in through a phone line, odds were those same phone lines were used to connect to their Internet Service Provider. Since people only had one phone line and one modem, they were either connected to the Internet or to the company network, but never to both at the same time. That, too, is no longer true. Legit employee access can be exploited, and your company firewall is not likely to notice that. Deploying some VPN technology may be the answer in this particular case, but make sure to have the right questions when you implement it. Securing just the connection between the home computer and your company network is not enough. Any computer that is connected to two networks at the same time is a potential relayer for attacks, and you need to make sure that the solution you choose addresses all possible threats.

There is a huge gap in security level between companies and families at home. Firewalls are traditionally expensive. Buying the firewall software and hardware and the machine to run it on, and training the system administrator in writing and maintaining the firewall policies (and in keeping the company policies on security in general up to date), takes money. Company firewalls usually allow people on the Internet to access Web and mail services provided by the company. It often also allows employees to access resources on the Web. Companies have to take the actions required to ensure that enough protection is in place to provide just those services, and those services only.

WHAT IS A FIREWALL ANYWAY?

A firewall is a computer that connects to two networks (usually the Internet and a local-area network) and doesn't allow any traffic from one to the other. Now, that can just as easily be achieved by not connecting the two networks together, and although the practical upshot of a firewall is that nobody on the Internet can connect to your shared printers and disk drives on the local network, it isn't really useful until you start adding things to it. Those things can be of two categories: allowing network packets of a certain kind to travel through, or acting on behalf of computers on one network to get some service from a computer on the other network. The firewall in this article is configured not to let any traffic through, and not to do anything on behalf of computers on the Internet side, and to act on behalf of the computers on the local network for any network service they want, through a mechanism known as NAT. Another way to do the same would be by using proxy applications. A proxy application knows one protocol only, such as Web requests, so you would need to add one for every service you want to provide. What the firewall does not protect against is malicious content. So if you pick up an email with a virus attachment, the firewall is not going to stop you. If that virus attachment is a tool that can be used to participate in the next DoS attack, the firewall is not going to stop that attack. However, nobody on the Internet who tries to trigger the attack can send a signal directly to the tool, because that would be stopped.

Home users have fewer requirements. Typically, they do not run a Web or mail server; they just want their home computer inaccessible from the Net, without being limited in their browsing.

However, even those lessened requirements come with a gotcha: the thousands of dollars a company normally invests in security are not available to the home user, so it is impossible to apply the same amount of effort that the company does. Acquiring the know-how is out as well – people have enough problems figuring out what the browser buttons do, or how to change the skin on the new version they just downloaded, and can't be bothered to learn about SYN floods as well. Therefore, there has to be an answer that does not cost a lot in terms of money or effort but meets the security requirements that are specific to those home users.

Shareware Answers

Lots of shareware authors jumped into this hole by writing little background applications running on Windows systems, trying valiantly to stop network services from being abused. Windows, however, is a moving target when it comes to protection on the computer itself, and besides, not everybody uses Windows. Mac and Linux owners face the same problems – they have to protect their systems from abuse, and the knowledge to secure those systems is not generally available. The rise of the ever more popular Linux is especially a problem. Sure, out of the box it is generally more secure than Windows, but since Linux is traditionally more a server system than a client, chances are people will, sometimes inadvertently, turn on those services. Telling people, "Well, don't do that then" is not an option – sharing Windows drives is far too useful to tell people not to do it, and Linux users typically want to play around with their systems. So, some time ago I set myself to developing a better answer.

WHAT TO AIM FOR?

I set myself a few limits. First, it had to run standalone. As any firewall administrator can tell you, "One function, one box" is an important design concept. It keeps things simple, and therefore easier to handle. Cost is another limit. If possible, whatever I came up with had to run on hardware already available, or, better yet, on surplus hardware. Since most cable-modem and ADSL owners are pretty computer-literate, probably having graduated from using dial-up connections, it is fairly safe to assume they have either an old computer lying around gathering dust or enough computer savvy to buy such a "boat anchor" cheaply. I decided my firewall had to be able to run on any discarded 80486 PC with less than 100MB of disk space, and 8MB of memory. Buying a second Ethernet card is just about the only cost I was willing to incur. From the feedback I got, these design decisions turned out to be well balanced. I've been told success stories by users who invested a few dollars in a secondhand computer; some bought only a second Ethernet card and a dust rag to clean out an old system.

I aimed at a "fire and forget" system. Of course, any company large enough to have a good security policy and dedicated staff will have to do maintenance all the time, but the home user probably wants to stick a box in the basement next to the cable entering the premise, and perhaps clean an air filter every two years, but that's it. Forget about it. Seemingly, this goes against the fact that security is an ever-continuing effort, but don't forget that this is a simple home-network solution. The ironic fact is, there are so many cable-modem and ADSL owners who don't secure their digital premises that even minor security measures are likely to send the cracker searching for easier targets. Not that I was going to settle for minor security improvements, but it's useful to keep in

I decided my firewall had to be able to run on any discarded 80486 PC with less than 100MB of disk space, and 8MB of memory.

The whole point of this particular setup is: Home users will never perform daily or even monthly maintenance, if any at all.

mind the kind of adversary that you're dealing with. Obviously, Amazon has different security requirements (and a budget to match) compared to Joe Six-Pack.

"Fire and forget" also has its effects on daily maintenance. For example, nobody is ever going to look at log files or console warnings. Sure, that's against every firewall policy I ever saw, but the whole point of this particular setup is: Home users will never perform daily or even monthly maintenance, if any at all. There is no sense in denying: yes, this is a firewall, but not as we know it.

Keeping the costs low means using existing open source wherever possible, and running on leftover PCs limits it to Linux or one of the BSD variants, since those operating systems still perform very well on those older machines. I picked NetBSD because that was the one I was most familiar with. It has to use NAT[2] so it can use one of the reserved address ranges[3] for the inside network. That way, people can configure their machines in any way they want, without having to worry about interfering with the Internet. Although NAT is not meant to be a firewall, using it with reserved addresses makes the internal machines effectively unreachable from the outside, and that's a very nice bonus. Talking bonuses, it also means that no matter how many computers there are behind the firewall, the ISP will see only one: the firewall. Since some service providers like to charge extra if more than one computer is hooked up to the connection, this could save people some money. It could be against ISP policy, but most providers only charge the extra money if you take up extra IP addresses.

Turning off all other network services on the firewall box and building a kernel that has unwanted or unneeded services turned off (such as source routing, debuggers, and optional filesystems like NFS) gives you a system that is already more secure than some commercially available firewalls. Curiously enough, the next step would be to write extensive packet-filter rules – something almost every firewall vendor will either do or require – but I decided against it in this case. It would be too much of an effort while adding relatively little extra security. Let's face it: most packet-filter rules are there to prevent address spoofing and to limit the availability of servers under specific circumstances. Since the typical cable-modem or ADSL owner is not likely to run servers (some ISP policies explicitly forbid it), one major reason for filtering goes out the window. Source routing could be another good reason for filtering, but I turned that off in the kernel. Some more exotic network features such as ICMP redirects and IP option flags remain, but they are usually of no relevance to this particular setup anyway. Again, remember that we're talking home networks here. By the time you get down to analyzing the risks of the things the filter rules protect you from, you're at a level far beyond what is required for this situation. There are simply no services offered by the firewall, viewed from the outside. There's no service listening to the network, and where there are no services, exploiting them is not really feasible. I've had some requests from people who do want to run services, most of them with small Web sites, and also some with networked multi-user games. When I find the time I will write some Web pages explaining how to add support for this to the firewall system.

INSTALLATION

Since we are talking "Fire and Forget" systems, the only remaining technical hurdle is the installation process. The NetBSD installer is already very friendly, but some assumptions about the installation can be made beforehand to simplify the installation process further. For example, the firewall box is going to run just one operating system, with a known set of software, so any disk partition and multiboot questions can be skipped, with appropriate defaults filled in. Adding support for DHCP is a must, since a

lot of service providers do not give their client fixed IP addresses. When you get down to it, you can almost reduce the questions you have to ask the user to, "What Ethernet card did you connect to the cable modem?" and "I am about to wipe your disk, is that OK?" How's that for simplicity?

Future Developments

Did I mention that any nonserving network can be reasonably protected by this scheme? SOHOs can use this as a way to protect their office networks as well – good security for a very low price and low overhead. In fact, from a technical point of view, even large businesses might find the system useful as a starting point. When you buy a new low-end Pentium system, you'll have at least 400MHz these days, with 64MB memory and a few GB of disk space – with a system like that, and the firewall installed, it is possible to support hundreds of users and a fat pipe to the Net. And all that without breaking a sweat. At that point, the focus shifts back to installing and maintaining services like mail and Web. I give some email support, but if a company wants to be able to fall back on the firewall provider 24 hours a day, I happily point them toward commercial firewall vendors. For me, this is just a volunteer effort, as so many open source projects are.

I admit that it has been very tempting to add services to the system. After all, Squid, Apache, and a mail daemon are easily added to the system, and before you know it, you've got a Small Business Server. However, I feel it wouldn't be a firewall any more, and I would not want to promote it as such. I might build a system like that at some point, since it's not difficult, but I would set it up as a different product, and serve it off a different domain. Way back in the good old days, a 4GB disk was something reserved for file servers. Today, a disk with many times that amount almost comes free with your breakfast cereal, so adding Samba and Netatalk to the Small Business Server is also a possibility. But I digress.

Conclusion

There's one important caveat to this story: I'm preaching to the choir here. Any one reading this magazine knows it's a jungle out there, and it's the average Web surfer at home who needs this information most. Getting the word out is not easy. Even the attacks on Amazon, CNN, and others in the beginning of 2000 go only so far in making people aware. Most people get a false sense of security, thinking it's something that affects Amazon, CNN, and a few other dotcoms, but not them. They don't know that it was the thousands of insecure computers (just like they have at home) that were used to launch the attack. I have yet to see statistics on what computers were used and what organizations own those computers. And it doesn't really matter anyway, because since those attacks took place the tools involved have migrated from the UNIX platforms they were exclusively running on to Windows computers (and, in one case, Macs). There are lots and lots more of those around. For a good time, take a look at <*http://www.dubbele.com*>, and remember to let me know what you think of it.

REFERENCES

1. <*http://www.clark.net/~roesch/security.html*>. Snort is a libpcap-based packet sniffer/logger that can be used as a lightweight network intrusion-detection system.

2. <*http://www.faqs.org/rfcs/rfc1631.htm*>. Network Address Translator.

3. <*http://www.faqs.org/rfcs/rfc1918.html*>. Address Allocation for Private Internets.

system administration research

Part 2: Analytical System Administration

by Mark Burgess

Mark is associate professor at Oslo College and is the author of cfengine and winner of the best paper award at LISA 1998.

<Mark.Burgess@iu.hioslo.no>

In my previous article (*;login:*, June 2000) I argued in favor of a more scientific approach to system administration. The key point was that we should be careful in making assertions without having something concrete to back them up. Also, I wanted to encourage more research of a scientific nature for LISA. In this follow-up article I want to look at some more concrete examples of how system administrators can get involved in research, both for the good of the community and as a discipline for making judgments on the job.

Research begins with a question: I wonder if . . . ? Or: Is it true that . . . ? Without such a question, you don't know what you are looking for. The danger of beginning with a question is that you then just set out to prove what you think is true, or disprove what someone else has said. The point of research is to get objective results, or report on ideas that stimulate progress in the field. We also need to take steps to check ourselves.

Everyone would like to think that they have an open mind, but that is not the way science works. I recall my time as a physics student at university in England. I recall writing that scientists are not really objective, but are human beings driven by opinions and ideas. My essay got a lousy grade from one of the lecturers: he proclaimed that scientists must be among the most objective people around. He was making a fatal mistake.

Self-made experts are almost always the least objective people around. They have worked hard to build up their knowledge, they have opinions, and they are pretty sure they know right from wrong without having to check. This kind of knowledge is based on experience, but take care not to be seduced by the dark side of the force! When you believe you know the answer, you might never find out that you are wrong. Science is not about being an expert, it is about being stupid, i.e. never assuming, always feigning ignorance, always being critical. Clearly it is easier to criticize or debunk than it is to make a constructive contribution, and many scientists have been seduced by that dark side (standing in the way of progress and confusing the issue with opinions rather than facts), but this is the challenge for a scientific community. The essence is to have a discipline that will lead to the right conclusion whether your mind is initially open or not.

Asking the Question

What kind of questions would a system administrator ask? Here are some examples:

Which is better, static mounting or auto-mounting?
In firewalls, how much does a proxy delay service availability?
Given identical hardware, which Web server and OS can be most efficient: Apache/Microsoft IIS, GNU/Linux, NT, FreeBSD?
Why is my system running more slowly than it used to?
Why does program X dump core every time it starts on one host, but not on another?

As you read these questions, you are probably already forming your own opinion about what the correct answer is. But what evidence do you have for your opinion? Long experience? A gut feeling? Hearsay? Let us look at the first of these. Which is better,

static mounting or auto-mounting? On reflection, we realize that this question has too many unknowns. So we try again:

Given a particular operating system, which is better?
But restricting to a single OS is not very interesting, since the results from one OS might actually be different from the results from another. A comparison of two would be the minimum we would accept as indicative.

When is static mounting more efficient than auto-mounting, or vice versa?
We now realize that we are in far more trouble than we realized. The question is potentially very complex. It will be necessary to consider a variety of cause-effect correlations in order to get a reasonable answer to my initially vague question. Then as we continue, we think of more things: What effect does environment have on whether the auto-mounter is better than static mounting? Which auto-mounter? Which static NFS implementation, TCP or UDP? Most important of all: What are we going to measure to find out the answer?

The Research Loop

Answering questions is a difficult business. It is a process:

```
#!/usr/bin/findout
#
#

ignorant=true;

while (ignorant || alive)
{
    Assess Motivation and Subject;
    DoMeasurements/Experimentation
    Interpret results
    Criticize interpretation
    if (results interesting)
    {
       Communicate results
    }
}
```

Notice how the loop doesn't end. Why not? Things change. An answer one day is not the right answer on a different day. Come to think of it: What constitutes an answer at all?

Take NFS: Does the question about static or auto-mounting have an answer? Does it have many answers, depending on time, place, environment, and so on? Can we ever say that the we have found the "right answer"? In the NFS example, there might not be a correct answer to the problem. Unless one can prove that one is intrinsically more efficient than another, and intrinsically more elegant, then most people would not care to ask such a broad question. The question should be restricted to the type: Under what circumstances is X more efficient than Y? This can be answered by measuring numbers.

The question should be restricted to the type: Under what circumstances is X more efficient than Y? This can be answered by measuring numbers.

Questions that ask us to make value judgments are very difficult to answer. Questions of this type can be discussed (this might be useful), but there is no right or wrong answer. The result of such a discussion can be used to motivate other studies, provide background knowledge for another study, or even demonstrate that a simple claim is in fact *not* true in general. However, nothing is proven to *be* true in general.

The most valuable kind of knowledge is the deeper understanding of why and how things happen. Understanding is usually about figuring out mechanisms that relate

The competition in research can be so intense that it becomes absurd. . . . You must make a judgment about the importance of earlier work.

cause to effect. Papers that increase understanding or awareness of actual phenomena are useful. Some questions are easy to answer, because the distance between cause and effect is small, e.g., Why does the file disappear when I type rm? Here the link is an atomic operation: delete. Other questions are harder to answer, e.g., Why does the system run slowly at certain times? Now there might be several causes to the observed effect. Elucidating the causal connection could be difficult. There are several approaches for doing this:

- Inspired guessing (followed by verification)
- Recognizing the signatures of known effects
- Gradual elimination of possibilities
- Statistical analyses (to test ideas or separate overlapping signals)

What Is Already Known?

There is no need to reinvent the wheel. It is both a waste of time and annoying to the original inventor. It is good practice to look at what has been done before, in order to avoid wasting time. It is also important to refer to what has been done before. The purpose of giving references is to place work in a context, to allow others to make the connections for themselves (for pedagogical reasons, and to verify your conclusions), and to avoid repeating what is already known. Tracking down references can be hard, and most work gets repeated in different contexts due to ignorance.

Naturally the inventor of a triangular wheel would like to be recognized for his/her work, but it might just confuse the issue. The competition in research can be so intense that it becomes absurd. Researchers in some environments are notorious for actively writing to authors of other papers telling them that they should be referred to. You must make a judgment about the importance of earlier work. On the one hand, you are not obliged to be the historian, summarizing the entire history of a subject on every occasion, but, on the other hand, you do need to tell the readers where you have come from and where you are going. Also, if others suggested the study you are making, it is important to refer to them: they thought of it first, and there was probably a reason why, relating to their own research.

For the NFS question, a quick search through LISA proceedings revealed three papers on NFS measurements and one on the auto-mounter. However, none of these dealt with comparing the efficiency of auto-mounter filesystems with static-mounted filesystems. This indicates that a study of this kind might be worthwhile. To confirm this hypothesis, I would then need to go and search through other journals, such as the ACM library or IEEE journals.

Measurements and Scales

Getting numbers is the most convincing way of making a point. A numerical value is less open to woolly interpretation than a descriptive result. To find numerics, you first have to find out what numbers can be measured and which of them, if any, are relevant to what you are trying to discover.

There are many sources of numbers. Typical monitoring commands for UNIX-like hosts, for instance, include:

```
ps
top
netstat
iostat (Solaris)
xload
perfmeter (Solaris)
```

These take snapshots of kernel values. The values need to be collected over time under similar conditions. If measurements cannot be made under similar conditions, the result will contain overlapping signals: one is the signal you are trying to measure; the other signals are background noise. So one thing to be cautious of in a multitasking system is that the existence of multiple processes implies multiple overlapping signals. If you are studying a single process, how are you going to separate the effect of the one process you are looking at from all of the others?

Several techniques are available. An understanding of scale can help here. One of the most important things to understand about dynamic systems, whose measurements change over time, is that very different things are going on at different scales. For example, put your hand in front of you and hold it still. At the scale of 1 cm your hand appears solid and still. If you swap your eye for a bionic appendage and zoom in (don't try this at home, kids) down to the sub-millimeter level, you see that there are all kinds of cellular things bubbling around and moving. Zooming in even further to the nanometer level, you see atoms flying around like crazy; further still, electrons going around in circles. Which picture is correct? Clearly there is information contained at every level, but the information concerns different aspects of the whole. The same phenomenon applies to any complex system. Computers are such a complex system.

For example, suppose we choose to measure disk usage. On a particular machine with no users but with some network services running, many temporary files are being created and destroyed, but on average nothing much happens. On average, the number of files does not change, since as many files are destroyed as are created. On the other hand, the number of bytes grows steadily. The astute experimenter determines that this is due to quietly growing log files, not to a leak in the network services.

If scale is important, how long do we have to measure something until we can be sure that we have seen what is really going on at all levels? How frequently do we have to measure in order to resolve an effect? Nyquist's sampling law says that, if we want to see effects on a time scale of t seconds, we need to sample at least every $t/2$ seconds. That is why CD recordings sample at 44 kHz, when humans can only hear up to about 20 kHz at best. Similarly, if we want to see an effect at scale t seconds, we need to sample for at least $4t$ seconds in order to be sure of seeing a whole cycle. These values are very rough; in general, you can never have enough data. You should sample much more than you think you need, just to be sure.

What overlapping signals might we see? Many computer measurements have a daily rhythm that is caused by the pattern of work of its users. It peaks around midday and is lowest during the night. This is a periodic signal that is mixed in with the general chatter of system behavior. Here cause and effect are easy to identify by plotting a graph of the data. It might be possible to see when users have their lunch break, just by measuring process behavior.

Be aware that the act of measurement can affect the measurement you are making. In order to make a measurement, you have to start a program that measures the system, but this uses resources too. Are those resources significant? For instance, if you run UNIX top to look at which process consumes most resources, and consistently see that top itself is the program that features highest in the readout, then you know that you are disturbing the system too much. This is a problem with any finite system. It is like the famous "uncertainty principle" in physics, sometimes called "Schrodinger's cat." The act of measurement might be the very thing that disturbs the system. To subtract the effects of measurement, we need to be able to control or predict their effect on the system.

Be aware that the act of measurement can affect the measurement you are making.

System administrators have many research skills already. Turning these skills into a research project takes only a little discipline.

Any meaningful result must be repeatable. If the result is not reproducible, it is of no value to science. Sometimes it is necessary to transform or manipulate data in order to find the features that are reproducible. It is not always the numbers that are reproducible, but their distribution or pattern of change. Finding the right variable or representation of data is a challenge for a researcher. This is part of what makes science fun. Even in something as simple as a desktop PC, there are things going on that are not easily appreciated without a little analysis. Finding out something that you hadn't realized, or something that confirms your suspicions, is a fantastically satisfying experience.

Resorting to Statistics

Statistics is about making the best of a bad lot. You don't have any clear idea what makes something happen, so you look to see what the laws of numbers can tell you. Statistics are used, broadly speaking, to separate signal from noise. More advanced notions of statistics have to do with determining relationships between cause and effect, subject to (i.e., filtered according to) certain conditions.

Statistical averaging is a little bit like half-closing your eyes to look at the data. When you deliberately blur an image, you see the main features more clearly, since distracting minor variations are blurred out. The technical term for this is coarse graining. The aim of averaging is to separate signal from noise. It is like the example of a hand, discussed above. If you always looked at your hand at the level of atoms, you would find it very confusing. However, if you change glasses and blur out the effects of individual atoms, averaging over individual cells so that your hand looks like a solid continuum, then it starts to make more sense as a hand. It becomes possible to understand its function.

In the example of temporary files above, we can safely say that the average number of files is a constant over long periods of time. Over short periods there are changes in the number of files. The average amount of disk space used rises gradually, however, due to the log files. Here we see how the process of averaging separates out behavior at different scales. The "error" or standard deviation is a measure of the average size of a short-time signal (often called a fluctuation).

Correlations can also be used to link cause and effect. There are auto-correlations, or correlations in a single measured value at particular times. This is a measure of how similar a value is at different periods of time. The correlation length (or time) is a measure of the distance over which the system seems to look uniform. Sudden changes in correlation length are referred to as phase transitions, after the same phenomenon in physics. They signal dramatic changes (called catastrophic changes) that imply a significant change in behavior. Cross-correlations measure how similar two separate sets of measurements are, i.e., whether it is likely that one measurement is affecting the other, or whether both signals have a common explanation. These are some of the tools that statistics has to offer for analyzing data.

Trial and Error

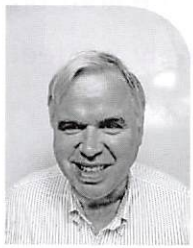
An important part of research is the ability to try and fail. One has to be willing to fail maybe 90% of the time in order to produce 10% of stuff worth telling someone about. The art of research is in channelling that 90% of failure back into the 10% of success, i.e., not just giving up on something interesting, but persevering until real progress is made.

System administrators have many research skills already. It is a part of the job to fiddle with stuff until the answer pops out. Turning these skills into a research project takes only a little discipline. The discipline is not wasted, even if does not always amount to a published paper.

I don't have the resources

by Steve Johnson

Steve Johnson has been a technical manager on and off for nearly two decades. At AT&T, he's best known for writing Yacc, Lint, and the Portable Compiler.



<scj@transmeta.com>

and Dusty White

Dusty White works as a management consultant in Silicon Valley, where she acts as a trainer, coach, and trouble-shooter for technical companies.

<dustywhite@earthlink.net>

Have you ever come up with a really good idea, one of those ideas that you just know will be great for the company, revolutionize your industry, or even make everyone's work easier? You could just feel that this was the right thing to do, and you could already see the results of having it implemented. You could even hear others already telling you that it's a great idea.

The only problem, just a minor problem in your eyes, was that you would need other people in the group to help you bring about the great things this idea promised.

So you rushed into your manager's office and poured out your idea, only to hear, "I'm sorry, we don't have the resources to do this."

What does this literally mean? If this idea were the most important thing for the group to do, would it get done? Certainly! It's clear that the immediate response translates to "No."

"No" is a very ambiguous word. It could well mean:

1. "I think this idea is really stupid, but I don't want to argue with you about it."

Or it might mean:

2. "I don't know how I'm going to get done what I already said I would do. Don't bother me with anything else."

Or it could mean:

3. "The other things my group is working on are all more important than working on your idea."

Or it could mean:

4. "I'm so stressed out I haven't really heard or understood your idea."

It's important to understand what the "no" means in order to decide how, or whether, to move it to a yes. How can you move beyond the no?

Several columns ago, we discussed chunking. Chunking refers to the size of the ideas being discussed – you can chunk up to bigger and bigger concepts by asking questions like "What is the intention of that?" or you can chunk down to smaller and smaller concepts by asking questions like "What, specifically?" We also pointed out that as you chunk up you tend to get agreement. (Listen to politicians on talk shows – they try to please everyone by using highly chunked phrases like "good government" and "compassionate conservatism," while their questioners keep asking "What, specifically?")

So when you get the "not enough resources" response, you might try chunking up. "Well, my idea is intended to support [lofty corporate goal] by [a few cogent details]." Hopefully, even if your manager thinks your idea is dumb, she/he will be nodding, since everyone supports [lofty corporate goal]. If your manager looks puzzled, expand on the response you get and give details until your idea is understood, including how it supports these corporate goals. Now, since you both support the same goals, it's natural to explore how the new idea could be supported and implemented. When you have agreement (and have eliminated the "dumb idea" response, above) you can now chunk down, preserving this agreement.

For example, you could ask, "What would happen if you supported my idea?" You could encourage your manager to get more detailed, and discuss who is working on

The work required to come to agreement on priorities is rarely wasted.

what in the group, and what would have to be dropped or delayed to support the idea. In this discussion, you might discover that all the people in the group are indeed working on things that are more important than your idea, and go off, with renewed respect for your manager's judgment, to look for another way of implementing the idea. Or you might discover that you could assist one of the group members in return for his or her help with your idea.

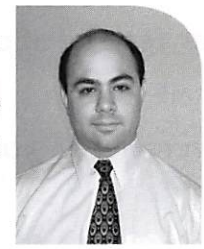
Even if the disagreement remains, by using this method you can usually reduce the impasse down to a priority call – is some current project more important than carrying out the new idea? You may need to seek help from outside the group in making this priority call.

The work required to come to agreement on priorities is rarely wasted. Some of the most frustrating business situations arise when two groups have different, unspoken rankings of company priorities, and proceed to frustrate each other while at the same time being convinced that they are safeguarding the best interests of the company. By smoking out and resolving these priority calls, many future problems can be avoided.

resume writing

A Hiring Manager's Perspective

by Christopher M. Russo



Chris Russo manages engineering at GTE Internetworking. His focus continues to be satisfying the customer – whether that be a Web developer, a system administrator, or an end user.

<chris@thlogic.com>

What is the mystery that lurks on the receiving end of resumes? Why is it that so many resumes we send seem to fall into some sort of black hole? Many of us are very qualified and do actually work very well with others – so why is it that no one seems to be banging down our door?

The answer? Your resume is terrible. I suppose you're thinking, "Oh, he means that most people have terrible resumes." Nope. You. You, reading this article right now – your resume is probably trash. I know mine is, and right after I'm done writing this article, I'm going to go fix that. Statistically speaking, very few people are likely to have "good resumes."

That's a pretty harsh, not to mention bold, statement. I don't make statements like that lightly, so I suppose I should explain.

As the title of this piece implies, I am a hiring manager. The group that I manage has literally the toughest job requirements that I have ever come across, and I have ten or more open requisitions that I need to fill. In the past six months I have reviewed hundreds and hundreds of resumes, and have spent a lot of time and effort refining my understanding of what a good candidate is and how to identify one from his or her resume.

I have done this by taking very careful notes on the resumes that I have reviewed, phone-screening candidates I felt might be appropriate, taking careful notes during the call, and then comparing my results to my original interpretation of the resume. In the beginning, I phone-screened almost every candidate whose resume came across my desk – a painful elimination process that took untold hours. Now, I phone-screen a precious few, and they are usually very close to my expectations.

In the interest of reducing the resume flow from 50 or 60 a day down to a slightly more manageable 10 or 15, I took this information and wrote it up so that I could provide it to my recruiters. Most of the ones who have taken the time to actually read and understand it have been hitting extremely close to the mark every time. The others I am systematically weeding out.

So now I am down to a point where I spend only about 30% of my time working on the hiring issues for my team, but I'm still looking to get more of that time back. I was trying to determine how I could improve this process even more, when I realized the only thing I could do would be to improve the quality of the resumes I was receiving, for even the resumes of the candidates who were qualified were still typically very poor and took a lot of extra time to parse.

I began detailing better resume-writing skills information to give to my recruiters, when I realized that the best way to communicate this information was to give it to the candidates. Thus, I decided to write this article.

It is my hope that this information will be helpful to you – and ultimately to me and other hiring managers. Understand that it is written with the assumption that if you are qualified, this will help you get more interviews, and much faster. If you are a charlatan and looking to get a job that you neither deserve nor are qualified for, then you probably need to go find an article on "creative resume writing," because this article deliberately defies such efforts.

Talk about the things you are proud of . . . the things that are very significant – major challenges you had to overcome in your career.

The following is a list of basic guidelines that I will use when rewriting my resume. I suggest you try them as well.

The Signal-to-Noise Issue

It is a good thing to keep in mind that, statistically speaking, most people who work in this industry are not really very good, and even fewer are capable of writing a good resume. Yours is likely to be swimming in a sea of resumes of this caliber. It is a difficult job showing your skills in situations like this, so be ready for the challenge.

Impressions

It is important to understand that resumes are all about impressions. If someone is looking to fill a position, then the likelihood is that your resume is one of dozens that they will review that day. Even the most well-meaning and conscientious managers are not going to read every word of every resume, because they simply do not have the time. It is therefore important that your resume quickly establish a positive impression of you and your abilities so that the person feels the need to investigate you further.

Sell Yourself

This probably sounds odd, given the nature of this article, but your resume is a showcase of the qualities that are you. Talk about the things you are proud of. If you are a technology analyst and a superstar, you shouldn't have things like "setup file shares" or "configured print queues" on your resume. It detracts from your other achievements and makes it appear as if you either didn't really do those more significant things, or just did them at a cursory level.

If you worked on some high-end projects, list those instead. Talk about the time you migrated the entire network to a new operating system, or the time you put in a business proposal to upgrade some major component on the 300 corporate infrastructure machines. Talk about the things that are very significant – major challenges you had to overcome in your career.

If you were an auto mechanic and writing your resume, do you think you would waste precious space on your resume explaining how you topped off the oil on a car and checked the tire pressure? One would certainly hope not. One would expect you would talk about the time you fitted that 350 big-block V8 into that 1958 MGB, or the time you fixed the transmission on a Ford Model T with only the aid of a screwdriver and a canned ham.

If it so happens that checking the oil and tire pressure were your greatest achievements, fine – put them down and be proud. Since, however, most of us have done a little more than that, most of us should be putting things of more import on our resume.

Objectivity

Don't put anything on your resume that isn't a fact. Statements like "excellent customer support skills" and "works well with others" are your opinions, and your opinions alone. These are the kinds of things that can only be proven by actually working in an environment, and it is foolish to think that just by writing them down that they will be taken as truth.

What you can do is detail experiences or job functions that might imply that you have these qualities. For example, instead of saying "excellent customer support skills," you could say "worked on the help desk for 12 months, during which customer satisfaction ratings went up 36%." Instead of saying, "works well with others," you might say something like "was delegated as liaison for all operations/helpdesk interdepartmental

issues." Both of these things will raise an eyebrow or two and may have the effect you were looking for. Be careful on the statistics, though – just because the helpdesk customer-satisfaction rate went up, that doesn't mean that you were the cause of it, and there is a very good chance you will be asked during your interview how you arrived at those numbers.

Honesty

Don't lie on your resume. In fact, don't even stretch the truth too much. The reality is that most good resume readers are going to be able to identify your tall tales fairly quickly. Your resume tells a story, and lies stand out as inconsistencies or deviations from the theme.

Yes, there are some people out there who will simply scan your resume for keywords, and if the keyword isn't there, they will throw your resume away – let them. If your resume is lacking something they are looking for, then there is no reason why you should want that job. There are far more jobs out there than there are people to fill them – you may not get that job, but you will most certainly get another one like it.

Length

Most resumes should be one or two pages long – in some cases you may need to make it three pages in order to be able to list all the technologies you understand and experiences you have had. If it's longer than that, I guarantee you that no one will read it unless they are very bored. They will read the first page or two, make their conclusions, and then throw it away.

If you can write a one-page resume, then you are a true master. I have actually seen extremely senior people write amazing one-page resumes. It's a tough thing to do, especially in a industry fraught with buzzwords and rapidly changing technologies, but it is possible.

On the other hand, if you really have only one page's worth of experience to write about, don't try to turn it into two. Some people want a more junior candidate and will be very annoyed with a resume from someone who is junior but trying to look senior by adding lots of "fluff."

Contractors are one of the few exceptions to the short resume. Most contractors have three, four, five, or more jobs each year, and keeping that history confined to a few pages is very difficult. For those people, I really don't know what to suggest, but I know that I still read the first couple of pages and ignore the rest. Further, eight-page resumes are very tempting to throw away just out of irritation.

The other exception is when you happen to need a curriculum vitae – in addition to their standard resume, many authors and other professionals have a much more extensive one that includes all of their experience and any publications they have been responsible for. This, however, is rarely required by the hiring party.

Keep It Concise

You do not need to tell your entire life's story in your resume. There was a time when I programmed in AppleSoft Basic, and I think I even have some experience with FORTRAN if I dig back far enough, but does anyone really care? And if they do, do you really want that job? Well, maybe you do, but I most certainly do not.

Your resume should detail the points in your experience set that are important to you and your career development. Make sure that what you list are things that you feel are

If you can write a one-page resume, then you are a true master.

It is important to keep your resume relatively generic, so that you won't be passed over for jobs that might be ideal for you.

either important to the jobs that you take or at least show something of significance in today's industry.

Listing too much simply confuses the people who are reviewing the resume and makes them question whether or not you really focus on the technology that they (and you) are looking for.

Keep It Generic

Your resume will be looked at by many different people – often with completely different jobs. Some jobs that may be perfect for you, but you may not have thought of them or even realized that they exist. It is therefore important to keep your resume relatively generic, so the you won't be passed over for jobs that might be ideal for you.

If you are feeling particularly zealous, you could theoretically slant your resume toward the needs of each specific job that you are applying for, but this is impractical for most people. Therefore, the focus should remain on the relatively generic.

Needless to say, don't go overboard here either. The opposite extreme is a single page with the words "did computer stuff" printed in the exact center of the page. This is clearly not useful either. Give them honest detail; the concern is just to watch that you don't paint yourself into a corner by giving too much.

Poor Documentation Skills

Wait a minute – you thought this was a piece on writing resumes, right? Well, yes, it is. Without question, if your resume is written poorly from the standpoint of grammar, spelling errors, or broken English, people are going to be left with a bad impression of you and your abilities. At the very least, they may think that you have poor documentation or presentation skills – both of which are important in many jobs.

Run your resume through a spell-checker, have someone review your resume for flow and good grammar, and if you have a difficult time with the English language, have a friend who speaks the language natively to review it and make suggestions for improvements.

Choose Your Words

Understand that resume writers often use words and phrases such as "participated in," "contributed to," or "assisted with" to attach themselves to a big project they really had little or no involvement in. Since this is a reasonably transparent scheme, these phrases have come to imply that the candidate not only was not a key contributor to the given task, but also is clearly trying to fool the reviewer into thinking that she or he was.

If, in reality, you were only "involved" in the task, then definitely list it as such. If, however, you did hold a more active role in the duty, be sure to use words that explain that. "Directed," "integrated," "migrated" are all active words that show you were the driving force in whatever task they are attached to.

In general, try to focus on things that you did personally, or rewrite statements to more clearly state what it was that you did. For example, instead of saying "was involved in the selection committee for conference abc," say "handled seating arrangements for selected officials at conference abc."

Paragraphs

Do not write your resume in paragraphs of text. This is not a novel you are writing, it is a resume. It should read like an outline, not like a biography. Paragraphs of text are hard to read, do not allow for easy skipping around, and leave you open to the reader disliking your style of writing.

Chronology

Dates are often very important to a resume reviewer. It is important to give your experiences in chronological order, listing most recent items first. This may seem obvious to most people, but I cannot tell you how many resumes I have seen that start with experiences from ten years ago. This can be very confusing and can leave a bad impression or, worse, can cause your resume to wind up in the wastebasket.

Many people today also perform several jobs at the same time – if you are one of those people, it is important to put "Contract" or "Part-Time" at the end of dates for jobs that overlap each other chronologically. This avoids raising questions in the mind of the reviewer.

Do not be afraid if you have lots of short-term jobs! Years ago, people looked for resumes that showed the person had devoted years to a company and was likely to stay on for the long haul. Interestingly, that trend seems to have reversed – now hiring managers are looking for people who are agile and can move quickly from job to job, because it is this kind of person who is often dynamic and able to change rapidly to meet the needs of a rapidly changing world.

It seems that now people who stay longer than 18 months are "lifers," and people who stay longer than two or three years are suspect. People expect a good system administrator to move on every so often, because a good admin learns fast and becomes bored very quickly. Unless you are holding down permanent jobs for three months at a time, detail your starting and ending dates with pride.

If you are holding down permanent jobs for only three months at a time, you may want to consider a different model entirely. Perhaps contracting or consulting would be more suitable for you – both are generally higher-paying careers and will be more apt to match up with your tendency toward mobility, and the short-term assignments will not look strange to hiring managers who review your resume.

Objectives and Goals

I have started cutting these "Objectives" out and hanging them on my wall so I can look up occasionally and get a chuckle on an otherwise tense day. All job objectives tend to boil down to the following statement: "Seeking a position where I can work insanely long hours and use my godlike skills to solve impossibly complicated problems while working hand in hand with people who love me dearly because I'm such an amazing person."

Oh please. These types of statements are neither honest nor objective. More often than not, they are just plain silly.

For the most part, I personally feel that objectives and goals are a waste of space unless you really know what you want, and it is fairly specific. If you genuinely know, for example, that you are "seeking a management position in the technology field, preferably in IT, secondarily in development" or something like it, then great! Those people who have the job that you are looking for may call you. Those people who are looking for a peanut vendor will not. Perfection.

It seems that now people who stay longer than 18 months are "lifers," and people who stay longer than two or three years are suspect.

Do not list every single technology you have ever been in the same room with. Listing skills or technologies implies that you can use them if called upon.

Those of you who decide to specify objectives need to understand that there is a very good chance that you might get passed over for a position you never would have thought of, but which might really interest you. Your objective may state "a," you may know that you don't want "b," but since "a" wasn't "c," your resume wound up in the trash. Perhaps this is a good thing, perhaps not. It depends on your goals.

If, however, you are like a lot of people who really are just looking for a cool job where you can poke around at a new technology, then leave the objective or goal off your resume entirely. You'll need the space for other things anyway, and "looking for a cool job where I can play with cool technology" is a little goofy and unprofessional-looking on your resume.

Summaries

Experience summaries are typically an attempt to give resume-scanners like myself a "quick overview" of the person's experience or skills. This is an interesting idea, but it is likely to be incorrect at best, at worst a major misrepresentation and disservice to the candidate.

Oftentimes recruiters encourage this behavior. Sometimes they demand it. As we stated previously, you are the person best suited to sell yourself, so the recruiter really shouldn't make a summary for you and is likely to represent you fairly poorly in any summary.

If you write the summary yourself, you will inevitably try to encapsulate what you are and what you can do in a one-paragraph statement. This is likely to lead to subjectivity and certainly will glaze over areas that may be key to one manager or another, thus causing your resume to be dismissed before even being fully reviewed.

For example, let's say that a manager is looking for scripting experience – even a small amount. Many system administrators will list on their resume a scripting experience they had, but are not looking for scripting jobs, so will not necessarily put it in their summary. What happens when your resume comes across the desk of a manager who will lazily read only the summary? Where do you think your resume will end up?

Skills

Many people list the technologies they are familiar with at the top of their resume. These lists are valuable for some people to do a quick check and make sure you at least have the basic skills they are looking for.

Do not, however, list every single technology you have ever been in the same room with. Listing skills or technologies implies that you can use them if called upon. If you are competent with the skill, then list it. If you just tinkered with it a bit one afternoon when you were bored, I don't recommend naming it. Think of it this way; you're going to wind up with either a very uncomfortable technical interview, or worse, a very uncomfortable day at some later time of explaining why your entire company went down because you couldn't fix something critical.

Another thing to consider when listing the skills is: Do you want to use this skill in your next job? For example, I have experience with, and can probably fix, a couple of nasty DOS-based accounting packages if I really want to . . . but I most certainly do not – not ever. Therefore, I usually think twice before putting it on my resume.

Real Experience

Don't try to pass off schoolwork, home projects, and fixing your neighbor's computer as "work experience." It's not. You could arguably list these things as technologies you

are familiar with, but unless you've done it in a professional environment where you were being paid for a measured deliverable, I don't recommend it. It makes you look silly and unprofessional, especially when you are asked about it during an interview. Picture yourself saying, "Oh yeah! My friend and I networked his house with some Ethernet cables we found over at the dump! It was cool!"

The only exception to this rule is if you happen to be a junior candidate, an intern, or someone who has recently graduated school and this is your first job. Use your best judgment.

Highlighting

Highlighting via bolding or italics is one of the curses of the overly specific method. The common practice is to bold or highlight key words or phrases to help resume-scanners like myself to see that you are perfect for the job.

There are many problems with this theory. The first is that it is very possible that you do not fully understand the job and what it would take to be perfect for it. Thus, you may highlight things that actually make you look inappropriate, or at least confused.

Another is that, as stated previously, your resume needs to be reasonably generic. How are you going to manage highlighting items that are appropriate for every job? Practically speaking, it isn't possible.

The last, and probably least obvious, is that it makes your resume very difficult, and very annoying, to read. It breaks up the flow significantly and causes whoever is reading it to stop and process why each bolded word has been so accented.

For example, try reading that last paragraph again, this time with key words bolded:

The **last**, and probably **least** obvious, is that it makes your resume very **difficult**, and very **annoying**, to read. It breaks up the **flow** significantly and causes whoever is reading it to stop and process why each **bolded** word has been so accented.

Recruiters

While a little outside the scope of this document, it is important to understand a few points when working with recruiters.

First and foremost is to understand that unless you have retained a recruiter to work for you – that is to say, you have offered payment to a recruiter to find you a position – the recruiter works for the company who eventually hires you. Remember, the recruiter gets paid a fee of 25–40% of your total salary if the company hires you. This means that the recruiter will always work in the company's best interest, and not yours.

Next it is important to keep in mind that the recruiters, in actuality, work for themselves. Just like you or me, they need to keep their own interests in mind at all times. There is a cost associated with helping you to find a position – if that cost ultimately outweighs what they believe that they will get in return when and if someone hires you, they may decide to give up on you.

This leads us to our next point – it is important that you work with the recruiter to help them find you a position. Be as forthcoming as you can with what your needs and expectations are, return calls promptly, and be professional in your dealings with them and the companies they send you to for interviews. The recruiter is your partner, albeit a limited one. The more assistance you give them, the more likely they will be to want to help you find a position.

Unless you have retained a recruiter to work for you . . . the recruiter works for the company who eventually hires you.

Any seasoned computer professional knows that you have certifications just so that people who are uninformed enough to demand them won't throw your resume in the trash.

RECRUITER "REVISIONS"

Recruiters have a nasty habit of modifying your resume to meet what they believe to be the requirements of the jobs that they are submitting you for. The problem here is that most recruiters do not understand your industry. Even those who basically understand cannot possibly keep up with the changes that happen almost daily in one of the strangest industries that ever existed.

Even if you are one of the worst tech professionals on earth, it is you who are most suited to sell yourself – not the recruiter. Make it clear to your recruiters that you have worked very hard on your resume to properly document your skills and capabilities and do not want your resume modified in any way. Be polite, of course, but be firm.

The only exception should be to allow your recruiter to put their company and personal information on your resume in the form of a header or footer. This is actually a good thing for the person reviewing the resume, as it allows them easily to pick up the phone and call the recruiter to request a phone screen or an interview.

Some recruiters may make suggestions on how you could improve your resume. If you respect their opinions, go with it. If not, tell them thank you, but you would rather keep it the way it is. If they insist, politely decline and find another recruiter. Don't worry about not being able to find other recruiters – you cannot sneeze without knocking over 50 or 60 of them.

OWN YOUR RESUME

Your resume, as a representation of yourself, is a commodity – and a valuable one at that. Most companies will pay a recruiter between 20 and 40 percent of your total compensation in commission to a recruiter who finds you. When you consider the salaries of tech professionals, you see that this is an enormous amount of money.

If multiple recruiters are presenting your resume to every single job opening in your area, managers are going to start seeing your name multiple times from different people. This causes confusion, and can even result in disqualification of your resume because companies are now uncertain as to who represents you. The last thing you want is for a manager to have a bad impression, or, worse, for you to lose a good job opportunity because you do not own the process.

It is important that you ensure that the recruiters who are handling your resume are not blasting it everywhere. As of this writing it is a job-seekers' market – most jobs are not filled for a very long time, because of the low supply of qualified talent. Make your recruiters work for their money. Tell them that they must contact you before submitting your resume to anyone, and, further, they must keep careful records on where your resume has been sent and who has sent it, as well as any notes on the progress of that submission so far.

Certifications

This section is likely to make me very unpopular, but I don't care – it must be said. Whether you believe it or not, most certifications are not really worth all that much. It's dangerous to make the assumption that your particular name-brand certification is worth more than some others and therefore assume that it is a critical piece of the resume. Unfortunately, the likelihood is that it is not.

Will these certifications get you a job? Possibly, but any seasoned computer professional knows that you have certifications just so that people who are uninformed enough to demand them won't throw your resume in the trash.

Other than that, they are typically the equivalent of a driver's test – proof only that you are not so insanely incompetent as to cause severe and consistent damage to others around you when at the console of the item in question. Certainly there are those among you who studied very hard and learned a lot – maybe even some who worked very hard to apply the knowledge after the fact – but those people are very rare. The vast majority of the people who have certifications have no real knowledge to back up their gilded paper.

This is unfortunate – especially for those of us who have the certifications and truthfully deserve them – but it's true. Once again, the signal-to-noise ratio is very poor.

If you have certifications, list them. Don't waste a lot of space – you need it for other more important things – put them all on one or two lines, and put them at the end of your resume. Do not splash logos on your resume. The more attention you bring to your certifications, the more it makes it look like you don't know what you are doing.

Of course, if you don't know what you are doing, please continue splashing logos all over your resume. It makes it much easier for managers like me to weed you out.

Education

A great many people in this industry have little or no college education – in fact, in my experience, many of the really good system administrators are the ones who flunked out of college because they were far too busy playing with computers to go to class.

This actually includes many people for whom you will be working. College educations do not appear to be a standard requirement, and frequently the only time they are listed in the job description is when someone in human resources put it there, or because the hiring manager is more "old-school."

If you have a college education, certainly list it – you worked hard for it and it does mean something. In addition, you don't want people passing you over just because you don't have something written on a piece of paper. It is also important to understand, that the higher up in the career band you go – manager, director, VP, CIO, CEO – the more important your degree is likely to be.

Do not, however, devote too much space to it. It is certainly far more significant than any certification you might have, but it is not typically nearly as important as your actual experiences. Again, typically list this at the end of your resume, but before your certifications.

Publications

If you have them, be sure to list them. Not many people write articles or books, and they do mean something – at least they mean enough to make someone stop and really think about how serious you are in your profession.

Personally, I prefer to find these at the end of a resume. When I see them closer to the top, I tend to worry whether the person is pompous.

"References Available Upon Request"

OK, did you really need to waste three lines worth of text and white space to tell me that if I ask for references you will give them to me? Please tell me you have more important things to put on your resume. Just take it off.

You need to have references, and I most certainly hope that you have some good ones, or you will have some rather serious issues. Aside from that, however, I think it is

The vast majority of the people who have certifications have no real knowledge to back up their gilded paper. This is unfortunate – especially for those of us who have the certifications and truthfully deserve them – but it's true.

Do the right thing; if you're right for the job, you'll get it.

assumed that you will provide references when requested, so you really don't need this on your resume.

So now that you have read all of this, I'm sure you will have found at least a few things that could use some adjustment on your resume. As I said, I know for a fact that mine needs a lot of fine-tuning.

If you'd like to try an interesting experiment, send out a couple of dozen copies of your resume – half of them before you make your changes, and half after. See how many interviews you get from each.

Remember that for a hiring manager looking for candidates, a resume is the sole window into the professional that is you. Make sure that it is something that you are proud to share with other people, and something that would never leave you feeling foolish or embarrassed. You are trying to get a job and establish a relationship with a corporation. If their first impression of you is that you are dishonest or trying to get a position you are clearly not cut out for, that impression is likely to bias their reactions in a negative way.

Do the right thing; if you're right for the job, you'll get it. If you're not, you'll get another one, and there's still always a chance that you will come back and get this job in another year or two. As I said, people move around a lot.

In the meantime, good luck, and happy job hunting!

isolation with flexibility

A Resource Management Framework for Central Servers

by David G. Sullivan

Faculty Advisor: Margo Seltzer, Harvard University

In managing computational resources, an operating system must balance a variety of goals, including maximizing resource utilization, minimizing latency, and providing fairness. The relative importance of these goals for a particular system depends on the nature of the system and the ways in which it is used. For supercomputers running compute-intensive applications, for example, the primary goal may be to maximize throughput, while for personal computers used to enhance a single user's productivity, the chief goal may be to maximize responsiveness.

In today's computing environments, users increasingly compete for the resources of server systems, whether to access central databases or to view content on virtually-hosted Web sites. On such systems, fairness becomes a critical resource-management goal. Proportional-share mechanisms allow this goal to be met by providing resource principals (users, applications, threads, etc.) with specified resource rights. For example, customers who pay Internet service providers to virtually host their Web sites can be guaranteed shares of the hosting machine that are commensurate with the price they pay. Service providers who can make such guarantees can offer larger resource shares to principals willing to pay a premium for better quality of service.

Although its full promise is yet to be realized, thin-client computing is another domain in which proportional-share resource management is desirable. Administrators of such systems are often forced to host one application per server to provide predictable levels of service [Sun98]. Proportional-share techniques enable the consolidation of multiple applications onto a single server by giving each application a dedicated share of the machine.

A system that supports proportional-share resource management must *isolate* resource principals from each other, so that a given principal's resource rights are protected from the activities of other principals. To provide such isolation, a system must necessarily impose limits on the flexibility with which resource allocations can be modified. Such limits work well when the resource needs of applications are well known and unchanging, because a system administrator can assign the appropriate resource shares and leave the system to run. Unfortunately, these conditions frequently do not hold. Even if the applications' current resource needs are adequately understood, they will typically change over time. For example, as a Web site's working set of frequently accessed documents expands, the site may require an increasing share of the server's disk bandwidth in order to offer reasonable responsiveness. Moreover, it would be preferable if system administrators could be freed from the need to make detailed characterizations of applications' resource needs. Ideally, the applications themselves should be able to modify their own resource rights in response to their needs and the current state of the system.

Our work provides extensions to the lottery-scheduling resource management framework [Wal94, Wal96] that allow resource principals to safely overcome the limits on flexible allocation that proportional-share frameworks impose for the sake of secure isolation. Our extended framework supports both absolute resource reservations (*hard shares*) and proportional allocations that change in size as resource principals enter and leave the competition for a resource (*soft shares*). It also includes a system of access

The USENIX Scholars Program provides support for student stipends, tuition and other expenses for students with exceptional research ability and promise. This and the following article are examples of the kind of work resultings from this support.

Applications may benefit from giving up a fraction of their resource rights for one resource in order to receive a larger share of another resource.

controls to protect the isolation properties that lottery scheduling provides. And our framework offers the means for applications to modify their own resource rights without compromising the rights of other resource principals. One of these mechanisms, called *ticket exchanges*, allows applications to coordinate their use of the system's resources by bartering with each other over resource rights. Our extended framework thereby provides *isolation with increased flexibility*: the flexibility to safely overcome the limits on resource allocation that standard proportional-share frameworks enforce.

Overcoming Upper Limits on Resource Allocation

The resource management framework developed for lottery scheduling is based on two key abstractions, *tickets* and *currencies*. Tickets are used to encapsulate resource rights. Resource principals receive resource rights that are proportional to the number of tickets that they hold for a resource; changing the number of tickets held by a resource principal automatically leads to a change in its resource rights.

Tickets are issued by currencies, which allow resource principals to be grouped together and isolated from each other. Principals funded by a currency share the resource rights allotted to that currency; currencies thus enable hierarchical resource management. Each currency effectively maintains its own exchange rate with a central base currency, and tickets from different currencies can be compared by determining their value with respect to the base currency (their *base value*). The more tickets a currency issues, the less each ticket is worth with respect to the base currency, and their total base value can never exceed the value of the tickets used to back the currency itself.

When a resource principal is funded by a currency other than the root currency, its resource rights can usually be increased by giving it additional tickets from that currency. However, because issuing more of a currency's tickets decreases their value, the resulting increase in the principal's resource rights is less than the increase in the number of tickets it holds. Indeed, no matter how many currency tickets a resource principal receives, the resource rights imparted by those tickets cannot exceed the overall rights given to the currency itself. This upper limit is essential to providing isolation. Without it, the resource rights of principals funded by other currencies could be reduced.

Despite the need for the upper limits imposed by currencies, these limits may often be unnecessarily restrictive. This is especially true on central servers, because the large number of resource principals a server must accommodate makes it difficult for a single allocation policy to adequately address their different and dynamically changing resource needs. Instead, some simple policy for ensuring fairness is likely to be used, such as giving users equal resource rights to divide among their applications, or allocating resource shares based on how much a user has paid.

Because certain resources may be more important than others to the performance of an application, applications may benefit from giving up a fraction of their resource rights for one resource in order to receive a larger share of another resource. We have therefore developed a mechanism called *ticket exchanges* that allows applications to take advantage of their differing resource needs by bartering with each other over resource-specific tickets. For example, a CPU-intensive application could exchange some of its disk tickets for some of the CPU tickets of an I/O-intensive application.

While ticket exchanges allow principals to obtain additional resource rights, they do so without compromising the isolation properties of the lottery-scheduling framework. Because the total values of the tickets outstanding for each resource are unchanged by

an exchange, only the resource rights of principals participating in an exchange are affected by it; the resource rights of nonparticipants are unaffected.

Ticket exchanges enable applications to coordinate with each other in ways that are mutually beneficial and that may increase the overall efficiency of the system. Various levels of sophistication could be employed by applications to determine what types of exchanges they are willing to make, and at what rates of exchange.

Certain types of resource principals may primarily need extra tickets for one particular resource. For example, consider two Web sites that are virtually hosted on the same server. Site A has a small number of frequently accessed files that it could keep in memory if it had additional memory tickets for its currency. Site B has a uniformly accessed working set that is too large to fit in memory; it would benefit from giving up some of its currency's memory tickets for some of A's disk tickets.

Applications could also apply economic and decision-theoretic models to determine, based on information about their performance (such as how often they are scheduled per second and how many page faults they incur per second) and the current state of the system (such as how many tickets of each type are active in the system), when to initiate an exchange and at what rate. This determination could be made by the application process itself, or by a separate *resource negotiator* process that monitors the relevant variables and initiates exchanges on the application's behalf.

Applications are free to cancel exchanges in which they are involved. This allows them to take a trial-and-error approach, experimenting with a variety of exchange rates until they achieve an acceptable level of performance, and to adapt their resource usage over time. Our initial experiments suggest that this type of approach will be necessary for most applications, because of the interdependencies that exist between different clients' resource usage.

Applications or their negotiators initiate exchanges by sending the appropriate information to a central *dealer*. The dealer maintains queues of outstanding exchange proposals, attempts to match up complementary requests, and carries out the resulting exchanges. If an exchange request cannot be satisfied immediately, the dealer returns a message that includes any proposals with conflicting exchange rates (e.g., process A requests 20 CPU tickets for 10 memory tickets, while process B requests 10 memory tickets for 10 CPU tickets). In this way, applications or their negotiators can decide whether to modify their proposed exchange rate and try again for a compromise deal. In environments where isolation is less important, the dealer could be modified to carry out exchanges that processes propose on the processes themselves (e.g., to take away 20 CPU tickets from a process and give it 20 extra memory tickets in return).

Future research is needed to develop negotiators suitable for a wide variety of applications and environments. Among the questions that still need to be addressed are: How can a negotiator determine what exchanges are beneficial to its associated process? When should a negotiator accept a trade less desirable than the one it proposed? Will a system involving dynamic ticket exchanges be stable (i.e., how can oscillatory behavior be avoided)? Can general-purpose negotiators be written that avoid the need to craft one for each application? In addition, the central dealer must be designed to deal fairly with requests that have complementary but differing exchange rates.

Applications are free to cancel exchanges in which they are involved. This allows them to take a trial-and-error approach.

To our knowledge, our prototype is the first implementation of a proportional-share framework to support both [soft and hard] shares for multiple resources.

Overcoming Lower Limits on Resource Allocation

Currencies can also impose lower limits on resource rights. These limits materialize when only one of the resource principals funded by a currency is in active contention for a given resource. In such circumstances, that principal receives *all* of the currency's resource rights, no matter how few tickets have been used to fund it.

As a result, currencies make it difficult for the lottery-scheduling framework to support the semantics of the `nice` utility found on conventional UNIX systems. For example, a user running a lengthy, CPU-intensive job may reduce its CPU funding as a favor to other users. But if the other tasks funded by the same currency are all idle, the CPU-intensive job will still get the currency's full CPU share. The user would presumably be allowed to decrease the number of tickets backing the user currency itself, but then other processes funded by that currency would also be affected when they became runnable.

While upper limits are necessary for providing isolation, lower limits are an undesirable side-effect of isolation. These limits could be overcome by funding CPU-intensive applications directly from the base currency. However, for reasons of security, access controls on currencies would presumably be used to prevent unprivileged users from issuing base-currency tickets.

To circumvent this restriction, our framework allows a user to issue base-currency tickets as long as the total value of currencies owned by that user never exceeds some upper bound. In this way, users can give up a small amount of their currencies' funding and then issue that same amount from the base currency to fund resource-intensive jobs. This approach leaves the user's total resource rights unchanged (preserving the isolation of other users), and the new job can run at a reduced priority without crippling the user's other applications.

Current Status

A paper that describes our initial ideas for extending lottery scheduling appeared in the 1999 HotOS Workshop [Sul99]. Since then, we have implemented the extended framework in VINO 0.50 ([Sel96], <*http://www.eecs.harvard.edu/~vino/vino*>), and used it to manage CPU time, memory, and disk bandwidth. A paper that describes our extended framework and its implementation in more detail appears in the 2000 USENIX Annual Technical Conference Proceedings <*http://www.usenix.org/events/usenix2000/technical.html*>. This paper includes descriptions of the allocation and scheduling mechanisms we have used in our prototype, along with experiments which demonstrate that they provide accurate proportional-share guarantees and effective isolation. It also presents results of experiments which test the impact of ticket exchanges on two sets of applications, demonstrating that the extended lottery-scheduling framework enables server applications to achieve improved performance under realistic usage scenarios.

Our work thus far makes several contributions. First, we have extended the lottery-scheduling framework to securely support the management of multiple resources, providing both soft and hard resource shares. To our knowledge, our prototype is the first implementation of a proportional-share framework to support both types of shares for multiple resources. Second, we have pointed out an important tension between the conflicting goals of secure isolation and flexible resource allocation—a tension that exists in any proportional-share resource management framework (e.g., [Ban99, Brun98]), not just lottery scheduling—and we have presented mechanisms that allow for more flexible allocation while preserving secure isolation. Third, we have illustrated

the value of a system that can support dynamic adjustments to the resource allocations applications receive.

In order for our extended framework to be fully effective on large central servers, more work needs to be done to develop negotiators that can intelligently carry out ticket exchanges on behalf of users and applications. We have recently begun work on this part of the project. Developing such negotiators will be a challenging task, but one with potentially significant rewards.

REFERENCES

[Ban99] Banga, G., Druschel, P., Mogul, J.C., "Resource Containers: A New Facility for Resource Management in Server Systems," *Proc. of the Third Symposium on Operating Systems Design and Implementation*, February 1999.

[Bru98] Bruno, J., Gabber, E., Özden, B., Silberschatz, A., "The Eclipse Operating System: Providing Quality of Service via Reservation Domains," *Proc. of the USENIX 1998 Annual Technical Conference*, June 1998.

[Sel96] Seltzer, M., Endo, Y., Small, C., Smith, K., "Dealing with Disaster: Surviving Misbehaved Kernel Extensions," *Proc. of the Second Symposium on Operating System Design and Implementation*, October 1996.

[Sul99] Sullivan, D., Haas, R., Seltzer, M., "Tickets and Currencies Revisited: Extending Multi-Resource Lottery Scheduling," *Proc. of the Seventh Workshop on Hot Topics in Operating Systems*, March 1999.

[Sun98] "Solaris Resource Manager 1.0: Controlling System Resources Effectively: A White Paper," *<http://www.sun.com/software/white-papers/wp-srm/>*.

[Wal94] Waldspurger, C.A., Weihl, W., "Lottery Scheduling: Flexible Proportional-Share Resource Management," *Proc. of the First Symposium on Operating System Design and Implementation*, November 1994.

[Wal96] Waldspurger, C.A., Weihl, W., "An Object-Oriented Framework for Modular Resource Management," *Proc. of the Fifth Int'l Workshop on Object Orientation in Operating Systems*, October 1996.

agentk: a toolkit for enhancing agent interfaces

by D. Scott McCrickard

College of Computing and GVU Center, Georgia Institute of Technology

<mccricks@cc.gatech.edu>

The USENIX Scholars Program provides support for student stipends, tuition and other expenses for students with exceptional research ability and promise. This and the previous article are examples of the kind of work resulting from this support.

Agentk is a USENIX-funded project intended to support the use of constant, cyclic animation in graphical user interfaces, particularly in autonomous programs, such as software agents, that may need to communicate constantly. Information on the Web changes frequently but irregularly, and while various programs can collect and process data from the Web, displaying it to the user in an informative but nonintrusive manner is still an important issue. The Agentk toolkit is designed to provide smooth, easy-to-control animation with minimal effort by application programmers.

The Agentk toolkit extends the Tcl/Tk scripting language to facilitate the incorporation of various graphical and motion-based effects into user interfaces. Primary among the effects are cyclic animations that allow large amounts of information to be constantly and repeatedly displayed in a small screen space. The cyclic animations include a fade effect that fades between blocks of text or blocks of graphics, a ticker effect that scrolls text or graphics horizontally across the screen, and a roll effect that scrolls information vertically. Each of these effects is incorporated in its own widget which provides a well-defined, familiar programming interface to it.

To help the end-user understand when and how the information has changed, the Agentk animated widgets support automatic markups and history-based shadowing. Automatic markups use changes in color or text style to highlight changes in the information, while history-based shadowing shows a previous state of the information in the shadow of the original. Both effects are triggered by a change in the information and remain in the display cycle for a programmer-specified number of iterations or duration of time. With these techniques, an end-user can know at a glance not only that information has changed but how it has changed as well.

The fade, ticker, and roll widgets are programmed in the same way as any other widget in Tcl/Tk: by specifying the widget type (ex. label, scrollbar, fade), a position in the display tree (.fader, .dialog.display), and possibly some options (-width 50, -fg red, -showhistory yes). The widgets include all options of the standard label widget that is used to display static text and images, but also include additional options that allow a programmer to control features such as multiple content locations, animation speed, markups, and synchronization between widgets. During execution, the Agentk widgets automatically tune the animation to ensure that it runs at the speed designated by the programmer, regardless of the processor speed and machine load. In so doing, the Agentk toolkit provides a programming interface that can be easily understood and used by Tcl/Tk programmers while shielding them from most of the details of generating animation.

By implementing Agentk entirely in Tcl/Tk, the platform independence of Tcl/Tk is maintained, and Agentk-animated widgets can even run within a Web browser using the Tcl plugin. Tcl/Tk was selected as the programming platform also because of its networking and string parsing capabilities, since most of the applications written using Agentk extract and process information from Web pages, networked machines, or other shared resources. A Tcl/Tk script can be written quickly in a handful of lines, allowing a programmer to move on to other tasks while the monitoring program runs smoothly in a corner of the screen.

The Agentk toolkit provides the means for including animation in user interfaces, but it was also important to us to explore when and whether it should be used. Animation in interfaces, most notably animated ads on Web pages, is often dismissed as intrusive and annoying. Our usability studies explored the distraction and effectiveness stemming from the use of fading and tickering animations while performing other tasks. We learned that in certain situations, animation does not result in degradation in performance on other tasks and in fact can communicate information effectively. Furthermore, it appears that different types of animation are better for different types of tasks. Use of the fading animation results in changes being noticed more quickly, while the tickering animation results in better future recall. These and other results can be used to guide designers based on the targeted goals of the application.

The primary design, implementation, and testing of the Agentk package was performed by Scott McCrickard, a Ph.D. candidate in the College of Computing and Graphics, Visualization, and Usability Center at Georgia Tech. Alex Zhao, also a Ph.D. candidate in the College of Computing and GVU Center, is responsible for the implementation of the image fading technique and for various performance enhancements. Richard Catrambone, a professor in the School of Psychology and GVU Center, provided assistance and advice on the usability testing. Scott and Alex's advisor is John Stasko, a professor in the College of Computing and the head of the Information Interfaces research group at Georgia Tech, a group dedicated to developing ways to help people understand information via techniques in information visualization, user interface design, and software agency.

Agentk has been available on the Web for over a year, and the latest release of the toolkit attracted several hundred visitors in the first few weeks. Since Agentk and Tcl/Tk are easy to learn, they have been used in short-term course assignments and summer projects, with impressive results. Agentk has been the topic of several papers, most notably one at the USENIX Tcl/Tk Conference in March which received the Best Student Paper Award. For more information about Agentk, including access to the most recent version of the toolkit, links to programs written using Agentk, and papers written about Agentk, visit the project Web site at *<http://www.cc.gatech.edu/~mccricks/agentk/>*.

The Agentk toolkit provides the means for including animation in user interfaces, but it was also important to us to explore when and whether it should be used. Animation in interfaces, most notably animated ads on Web pages, is often dismissed as intrusive and annoying.

USENIX news

Message from the President

by Daniel Geer

President, USENIX Board of Directors

<*geer@usenix.org*>

"USENIX has, in many ways, made me what I am today, and it is with humble gratitude that I accept the chance to put back into this very special organization a small measure of what I owe it, what I owe you collectively." That is what I said in my candidate statement. I meant it then; I mean it now.

Folks, this is your organization. What you get out of it is, in the cornball truth, what you put into it. We don't care what you know or do, we can find you a way to use it and to grow with it. This is not about miracles, it is about a world that clearly values initiative, current knowledge, and a web of colleagues that you can think of as your own partners. That is why I quoted my own damned candidate statement to start off my maiden article as President.

I'm writing this on 23 May. By the time you read it, the new Board of Directors will have met, we'll have had our first fist-fight over what order to do what, and then divvied up the work that goes with it. The Board is made of just the sort of people somebody was thinking of when they supposedly said, "When you have to get something done, give it to the busiest person in the shop – they may be busy but the reason they are busy is that they get the most things done." This group is the ghost of USENIX future; if you are interested in that future, ask them how to help. Trust me on this – the way to become an expert in your field is to start acting like one, and a professional association as good as USENIX is the densest opportunity space you're likely to stumble over. Want to look like a genius? Know the future. Want to know the future? Be part of putting on a meeting where the future gets announced. Think you know where the future is? Put a paper into one of our meetings or, if is a Really Big Idea, talk to the Board about it, since we regularly stage workshops on new topics. Like teaching so much that above all else you like your audience to ask hard questions? Put in a tutorial proposal. Today. First-time attendee? Find a

way to come back, and if your current employer doesn't want to invest in you that much, all the more reason to do it yourself.

Tired: The Big eat the Small.

Wired: The Fast eat the Slow.

All Aboard!

Balkan Olympiad in Informatics

by Don Piele
USA Team Leader
<*piele@cs.uwp.edu*>

USA team members Reid Barton from Arlington, Massachusetts, and John Danaher from Springfield, Virginia, finished number one and number two at the 8th Balkan Olympiad in Informatics, which took place in Ohrid, Macedonia, on May 15-19. They received the top two of the four gold medals awarded at the event.

Team members Percy Liang from Phoenix, Arizona, and Yuran Lu from Presque Isle, Maine, finished 13th and 18th respectively and received bronze medals. The USA team's combined score ranked second, behind the team from Romania which captured the third and fourth gold medals and two silver medals.

The USA team participated as a guest for the first time in the Balkan Olympiad, which this year attracted teams from nine countries: Romania, Yugoslavia, Bulgaria, Greece, Cyprus, Albania, Georgia (another guest), and the host country, Macedonia.

Marjan Gusev and his staff of volunteers from the Institute for Informatics in Skopje, Macedonia, worked around the clock to put on a very enjoyable and successful program. Unlike the larger International Olympiads in Informatics, which involve 65 countries, the Balkan Olympiad was a more personal event, with lots of opportunities to socialize with other team leaders and members of Marjan's staff.

The final rankings are posted on the BOI web site <*http://www.uwp.edu/academic/mathematics/usaco/2000/BOI/boi.htm*>.

The USA Team is sponsored by USENIX.

MEMBERSHIP
Telephone: 510 528 8649
Email: <*office@usenix.org*>

PUBLICATIONS/WEB SITE
<*http://www.usenix.org*>
Jane-Ellen Long <*jel@usenix.org*>
Telephone: 510 528 8649

SAGE news

What's in a Name?

by Barbara Dijker

Barbara Dijker is currently SAGE president. She's been sysadminning for about 12 years and runs a couple of ISPs.

<barb@usenix.org>

Recently SAGE has been trying to get more exposure for the organization and system administration. In doing so, I've been talking to many folk who don't run in our crowd. I've run into something I didn't expect.

It would seem that to those coming from the Microsoft side of the universe, the term "systems administration" is used only to refer to UNIX systems administration. Those whom we would call system administrators of Microsoft systems actually call themselves network administrators.

I find this confusing. In our view of the world, network administrators have nothing to do with Microsoft. They're the folks who work exclusively on our LAN or WAN, being responsible for managing IP address space, router configuration, setting up VLANs, etc. Sometimes we as system administrators do this too, so we might say "system and network administration." We do that to be inclusive of network-specific tasks, not inclusive of Microsoft. Many folks I would call network administrators call themselves network engineers (probably when they get their CCIE), and they insist it would be inappropriate for us to call ourselves system engineers.

At first I thought maybe this was just a one-off, the perspective of a lone confused person. But I've heard it over and over again. A Microsoft admin came to one of our SAGE local group meetings. The terminology was a source of confusion as a result, because the topic was system administrator salary survey results and included NT administrators. I also pinged some folks about it while at the SANE conference in the Netherlands. The same discrepancy is apparent outside the U.S. as well. The most recent jobs survey from the Information Technology Association of America (ITAA) doesn't identify system administration as a job category, because they use network administration. Supposedly this is a "standard" they got from the North-west Center for Emerging Technologies (NWCET).

I'm also at a loss as to where this terminology came from. Is it a remnant from the Novell days? Is it Microsoft spin? Is it because the term "system" has been used more with mainframes than with desktop workstations and workgroup servers? Is it Microsoft's emphasis on network services, e.g., you log into the network, not the server?

Should we care? If we really think that "systems administration" as a profession is independent of architecture, then we should care. In the entire history of SAGE, I recall no intention of being exclusive to UNIX. In fact, every year SAGE discusses how better to reach out to our Microsoft brethren. The terminology makes a tangible difference in little things like trying to reach the right audience for events such as the LISA-NT conference. The name alone tells those from the Microsoft universe that it is a conference for UNIX system administrators who are forced to deal with NT. That's of course not the intention of the conference at all.

I'm not quite sure what to do about this. Continuing what we have been doing can't hurt: raising awareness of SAGE and systems administration as we see it, with no barriers. Until we can figure out a good way to resolve this discrepancy, SAGE will be referred to as SNAG. ;-)

SAGE, the System Administrators Guild, is a Special Technical Group within USENIX. It is organized to advance the status of computer system administration as a profession, establish standards of professional excellence and recognize those who attain them, develop guidelines for improving the technical and managerial capabilities of members of the profession, and promote activities that advance the state of the art or the community.

All system administrators benefit from the advancement and growing credibility of the profession. Joining SAGE allows individuals and organizations to contribute to the community of system administrators and the professions as a whole.

SAGE membership includes USENIX membership. SAGE members receive all USENIX member benefits plus others exclusive to SAGE.

SAGE members save when registering for USENIX conferences and conferences co-sponsored by SAGE.

SAGE publishes a series of practical booklets SAGE members receive a free copy of each booklet published during their membership term.

SAGE sponsors an annual survey of sysadmin salaries collated with job responsibilities. Results are available to members online.

The SAGE Web site offers a members-only Jobs-Offered and Positions-Sought Job Center.

SAGE MEMBERSHIP
<office@usenix.org>

SAGE ONLINE SERVICES
List server: <majordomo@sage.org>
Web: <http://www.usenix.org/sage/>

Miscellaneous News

Progress on the SAGE Certification Project is continuing. An expanded SAGE Certification Policy Committee will be meeting August 1, 2000, in Seattle, Washington, USA, to make some decisions about the program that are necessary preliminaries to the development of the exams.

A successful SANE (2nd International System Administration & Networking) Conference, sponsored by NLUUG, USENIX, and NLnet, was held in Maastricht, The Netherlands, in May. It was attended by over 600 system administrators from almost 30 countries. Conference summaries will be published in a future issue of *;login:*.

The Computing Research Association (CRA) Conference at Snowbird 2000 is being held July 9–11, 2000. This is a conference of chairs of university computer science and computer engineering departments. At this conference, David Parter of the SAGE Executive Committee will be chairing a session on systems administration courses and the CS curriculum. The session will cover background on the field; existing systems administration courses; curriculum issues, including integration into the existing CS curriculum; and other issues likely to be encountered in creating and sustaining a systems administration course. Speakers include David Parter (University of Wisconsin and SAGE), Jerry Neece (Sun Microsystems), and Evi Nemeth (University of Colorado). For more information about the event, see *<http://www.cra.org/Activities/Snowbird00.html>*.

SAGE is expanding its outreach by increasing its presence at non–SAGE-sponsored events. You can help by coordinating a SAGE BoF or talk at a non-USENIX conference or workshop in your area. For support information and literature, contact *<gale@usenix.org>*.

SAGE is initiating a new project geared toward students interested in systems administration. This program, the Student SysAdmin Internship Program, links administrators of university computing environments with students who want on-the-job experience. For more information, contact Peg Schafer *<peg@usenix.org>*.

USENIX has established a new international research exchange program, ReX (Research Exchange). System administrators should apply. Contact *<rex@usenix.org>* for more information.

The SAGE Executive Committee will meet next during the LISA-NT conference in Seattle, Washington, USA, on July 31, 2000. We hope to see you there.

Growing SysAdmin as a Profession: Local Groups

by Bruce Alan Wynn

Bruce Alan Wynn is a member of the SAGE STG Executive Committee.

<wynn@sage.org>

One of the foundations of the SAGE charter is the desire to advance the status of computer systems administration as a profession. One of the efforts to accomplish this is the sharing of information about our profession among our members. This is accomplished in the "hallway track" at conferences, on the "sage-members" mailing list, and by sharing information locally within a given geographical area.

Not everyone can attend conferences to participate in the "hallway track," and not everyone can benefit from a mailing list with thousands of members. Most people, however, can participate on a smaller scale and contribute to the larger effort.

A Local Group is simply a group of system administrators in a given geographical area who meet periodically to exchange information about technology, tools, events, job openings, and anything else related to our chosen career.

Why form a local group?

In any area, there are probably a number of system administrators who could benefit from the sharing of information. It is not always clear, however, who those people are. A local group provides a common meeting ground that is not tied to a specific employer or vendor, where technical professionals can meet at a predefined time and location.

Further, if you're reading this you're likely already a SAGE member. But do you know who the other SAGE members are in your area? Do you exchange information with them? Do you promote SAGE when you talk to other sysadmins? A local group can do all of these things for you!

What is a SAGE local group?

Any sysadmin group can be affiliated with SAGE. The benefits of this affiliation include limited rights to use the SAGE name and logo, listing of the local group on the SAGE Web site, and SAGE mailing list support. The group must maintain a purpose consistent with that of SAGE – generally speaking, to advance the systems administration profession.

Who can participate in a SAGE local group?

SAGE local groups must maintain an open membership, allowing membership to any system administrator in the defined geographical area. This means that meeting locations must be open to the general public, not within a secured facility.

Who can form a SAGE local group?

The founding of a SAGE local group requires a minimum of two SAGE members. If you need assistance in finding additional SAGE members in your area, either for the founding of the group or to notify local members of the group's formation, you can get assistance by sending email to *<office@usenix.org>*.

Other details of SAGE local group requirements are available on the SAGE Web site and include:

- Represent a defined geographical area;
- Meet at least four times per year;
- Not organized for profit;
- At all times have at least two members who are members of SAGE (designated sponsors).

How can I form a SAGE local group?

Forming a SAGE local group is very easy. Proposals requesting recognition as a local group should reflect the criteria described above and should be sent by email to Gale Berkowitz *<gale@usenix.org>*.

For more information about SAGE local groups, please see *<http://www.usenix.org/sage/locals/sage-localgroups.html>*.

Announcement and Call for Papers **USENIX**

2001 USENIX Annual Technical Conference

http://www.usenix.org/events/usenix01

June 25-30, 2001 **Marriot Copley Place Hotel, Boston, Massachusetts**

Important Dates

FREENIX Refereed Track submission deadline: *November 27, 2000*
General Session Refereed Track submission deadline: *December 1, 2000*
Notification to authors: *January 31, 2001*
FREENIX Track papers due for shepherding: *April 16, 2001*
Camera-ready papers due: *May 1, 2001*

Conference Organizers

Program Committee:
Chair: Yoonho Park, *IBM Research*
Mohit Aron, *Rice University*
Carla Ellis, *Duke University*
Wuchi Feng, *Ohio State University*
Greg Ganger, *Carnegie-Mellon University*
Sheila Harnett, *IBM Austin*
Peter Honeyman, *University of Michigan*
Jochen Liedtke, *University of Karlsruhe*
Robert Miller, *Carnegie-Mellon University*
Vern Paxson, *ACIRI*
Doug Schmidt, *University of California, Irvine*
Margo Seltzer, *Harvard University*
Dan Wallach, *Rice University*

Invited Talks Coordinators:
Matt Blaze, *AT&T Labs–Research*
John T. Kohl, *Rational Software*

FREENIX Program Committee:
Chair: Clem Cole, *Compaq*
Ken Coar, *Apache*
Chris Demetriou, *NetBSD*
Ted Faber, *ISI*
Drew Gallatin, *Duke/FreeBSD*
Alan Nemeth, *Compaq*
Simon Patience, *SGI/Linux*
Garry Paxinos, *XFree86/Metro-X*
Stephen Tweedie, *RH/Linux*

Overview

USENIX is the Advanced Computing Systems Association. For over 25 years, its members have come from a broad community of developers, researchers, system administrators and engineers with interests spanning the full range of technology. As the core conference of this community, the USENIX Annual Technical Conference is the premier forum for computing professionals to share the results of their latest and best work, develop new ideas and solutions, and connect with their colleagues.

Three days of tutorials start the conference with practical tutorials on timely topics. The three-day technical session of the conference follows and includes a track of General Session Refereed Papers selected by the Program Committee; a track of Invited Talks by experts and leaders in the field; and FREENIX, a track of refereed papers on freely available POSIX-based software and systems.

General Session Refereed Papers

The 2001 USENIX Technical Conference seeks original and innovative papers about the applications, architecture, implementation, and performance of modern computing systems. As at all USENIX conferences, papers that analyze problem areas and draw important conclusions from practical experience are especially welcome. Some particularly interesting application topics are:

- Cluster computing
- Complexity management
- Distributed caching and replication
- Extensible operating systems
- File systems and storage systems
- Interoperability of heterogeneous systems
- Mobile code
- Networking and network services
- Multimedia
- Pervasive computing
- Reliability and QoS
- Security and privacy
- Web technologies

As at all USENIX conferences, papers that analyze problem areas, draw important conclusions from practical experience, and make freely available the techniques and tools developed in the course of the work are especially welcome.

Cash prizes will be awarded to the best papers at the conference. Please see the Web site for examples of the 1999 Best Papers.

How to Submit a Paper to the General Session Refereed Track

Authors are required to submit full papers by Friday, December 1, 2000.

All submissions for USENIX 2001 will be electronic, in PostScript or PDF. A web form for submissions is available on the conference Web site.

Authors will be notified of receipt of submission via e-mail. If you do not receive notification, contact: *usenix01chair@usenix.org.*

Papers should be 8 to 12 single-spaced 8.5 x 11 inch pages (about 4000-6000 words), not counting figures and references. Papers longer than 14 pages and papers so short as to be considered extended abstracts will not be reviewed.

It is imperative that you follow the instructions for submitting a quality paper. Specific questions about submissions may be sent to the program chair via email to: *usenix01chair@usenix.org.*

A good paper will clearly demonstrate that the authors:

- are attacking a significant problem,
- are familiar with the literature,
- have devised an original or clever solution,
- if appropriate, have implemented the solution and characterized its performance using reasonable experimental techniques, and
- have drawn appropriate conclusions from their work.

Note: the USENIX Technical Conference, like most conferences and journals, requires that papers not be submitted simultaneously to more than one conference or publication, and that submitted papers not be previously or subsequently published elsewhere. Papers submitted to this conference that are under review elsewhere will not be reviewed. Papers accompanied by non-disclosure agreement forms can not be accepted, and will not be reviewed. All submissions are held in the highest confidentiality prior to publication in the Proceedings, both as a matter of policy and in accord with the U.S. Copyright Act of 1976.

Authors will be notified by January 31, 2001. Some accepted papers will be shepherded by a program committee member through an editorial review process prior to final acceptance for publication in the proceedings.

FREENIX Refereed Track

FREENIX is a special track within the USENIX Annual Technical Conference. USENIX encourages the exchange of information and technologies between the commercial UNIX products and the free software world as well as among the various free operating system alternatives.

FREENIX is the showcase for the latest developments and interesting software applications in a form that is being freely redistributed. The FREENIX forum includes Apache, FreeBSD, GNOME, GNU, KDE, Linux, NetBSD, OpenBSD, Samba, and more. The FREENIX track attempts to cover the full range of software which is freely redistributable in source code form and provides pointers to

where the code can be found on the Internet. Clem Cole is serving as FREENIX program chair.

Submissions to the General Session Refereed Track are expected to represent mature work for which the authors are ready to fully describe the background, new ideas, experiments, and results of their work. By contrast, the FREENIX Track seeks to gather reports on projects that are currently and solidly underway, but may not yet be completely finished. This differs from a Works-In-Progress session which is really a poster for ideas.

FREENIX is looking for paper about projects with a solid emphasis on nurturing the open source/freely available software community. The purpose for the FREENIX papers is not as wholly an archival reference, but rather a place to let others know about the project on which you are working and to provide a forum from which to expand your user base.

We are looking for talks which advance the state of the art of freely redistributable software or otherwise provide useful information to those faced with deploying (and selling) free software in the field.

Areas of interest include, but are not limited to:

- Operating system design
- Network design and implementation
- File system design
- Highly-available systems
- Highly-scalable systems
- Graphical user interface tools
- Desktop metaphors
- File and print systems
- System management tools
- Security
- Large scale system management
- Interesting deployments of free software
- How free software is being developed and managed today

Interesting applications of freely redistributable software might include: robotics and automation, clustering, wearable computers, embedded systems, high-speed networking, studio graphics, and audio processing.

Cash prizes will be awarded for the best paper and the best paper by a student.

How to Submit to the FREENIX Refereed Track

Authors are required to submit one to three page summaries by Monday, November 27, 2000.

All submissions for the FREENIX Track will be electronic. A Web form for submissions is available on the conference Web site.

Authors will be notified of receipt of submission via e-mail. If you do not receive notification, contact: *freenix01chair@usenix.org.*

You may submit a full paper, however we expect that most submissions will be 1–3 page summaries of your work to date. Please provide enough detail to let us know what you are doing. In no event should you submit a description in excess of 14 pages.

Specific questions about submissions may be sent to the program chair via email to: *freenix01chair@usenix.org.*

A good submission will clearly demonstrate that the authors:

- are attacking a significant problem that has general interest to the Freenix community,
- are actively working on a solution, and
- have made enough progress to have useful information to report in a formal paper.

In particular, the FREENIX track is not for people that are thinking about doing some project, but have not yet started it. Such talks are better presented in the Work-In-Progress (WIP) session.

Authors will be notified by January 31, 2001. All accepted authors will be expected to produce a written report for the proceedings. These reports must be reviewed and accepted by the paper's shepherds by April 16, 2001. After shepherd approval, the reports then must be submitted with the standard release forms to the USENIX office by May 1, 2001. If you would like to avoid future formatting changes, you may consult predefined templates on the conference Web site.

These reports need not be as polished papers as would be submitted to the general session refereed track, although the higher quality submissions are often better received by the community. The papers should describe work that has been completed as of the time of their submission. The purpose of your paper is to let readers and attendees know what you are doing. Your talk at the conference may describe not only what is in your paper but also the work completed between the time that the paper is submitted and the conference is held.

Submitting a Tutorial Program Proposal

On Monday-Wednesday, June 25-27, USENIX's well-respected tutorial program offers intensive, immediately practical tutorials on topics essential to the use, development, and administration of advanced computing systems. Skilled instructors, who are hands-on experts in their topic areas, present both introductory and advanced tutorials covering topics such as:

- High availability and quality of service
- Distributed, replicated, and web based systems
- System administration and security
- Embedded systems
- File systems and storage systems
- Interoperability of heterogeneous systems
- Operating systems (Linux, BSD*, NT, etc.)
- Application development (threads, Perl, etc.)
- Intrusion detection and prevention
- Internet security
- Mobile code and mobile computing
- New algorithms and applications
- Systems application configuration and maintenance
- Personal digital assistants
- Security and privacy
- Web-based technologies

To provide the best possible tutorial slate, USENIX continually solicits proposals for new tutorials. If you are interested in presenting a tutorial, contact: Dan Klein, Tutorial Coordinator, Phone: 1.412.422.0285, Email: *dvk@usenix.org*

Submitting an Invited Talk Proposal

These survey-style talks given by experts range over many interesting and timely topics. The Invited Talks track also may include panel presentations and selections from the best presentations at recent USENIX conferences.

The Invited Talks coordinators welcome suggestions for topics and request proposals for particular talks. In your proposal state the main focus, including a brief outline, and be sure to emphasize why your topic is of general interest to our community. Please submit via email to *usenix01it@usenix.org.*

Work-in-Progress Reports

Do you have interesting work you would like to share, or a cool idea that is not yet ready to be published? The USENIX audience provides valuable discussion and feedback. We are particularly interested in presentation of student work. To request a WIP slot, send email to *usenix01wips@usenix.org.*

Birds-of-a-Feather Sessions (BOFs)

The always popular evening BOFs are very informal, attendee-organized gatherings of persons interested in a particular topic. BOFs may be scheduled at the conference or in advance via email to *conference@usenix.org.*

USENIX Exhibition

In the exhibition, the emphasis is on serious questions and feedback. Vendors will demonstrate the features and technical innovations which distinguish their products. For more information, please contact:
Dana Geffner, USENIX Exhibition Manager
Phone: 1.831.457.8649
Email: *dana@usenix.org*

Registration Materials

Complete program and registration information will be available in March 2001 on the conference Web site. The information will be in both html and a printable PDF file. If you would like to receive the program booklet in print, please email your request, including your postal address, to: *conference@usenix.org.*

Announcement and Call for Papers

NordU2001 - The third NordU2001/USENIX Conference
February 12 – 16, 2001, Norra Latin, Stockholm, Sweden

Information regarding The third Nordic EurOpen/USENIX Conference, to be held at Norra Latin, Stockholm, Sweden, February 12-16, 2001

A conference organized by EurOpen.SE - The Swedish Association of Unix Users, and an affiliate of USENIX, the Advanced Computing Systems Association.

Important Date
Extended abstracts due September 8, 2000
Notification of acceptance October 10, 2000
Final papers due December 8, 2000

Authors are invited to submit a 1/1 page abstract in English on any of the topics below to the Congress Secretariat. Submission should be original work and will be reviewed by the Technical Programme Committee.
All accepted papers will be available on Internet after the Conference. Authors must register for the confernce and present their papers in person. Instructions for preparing final papers will be sent with the letter of acceptance.

Complete programme and registration information will be available by mid October 2000. To receive information about the third NordU/USENIX Conference, please visit **http://www.nordu.org/NordU2001/** or send an e-mail to **NordU2001@europen.se**

TOPICS
- Security
- Operating Systems
- Open Source/Free Unix
- Mobile Computing
- Software Development
- Interoperability
- Storage Area Network, SAN

Technical Programme Committee:
Jan Säll, EurOpen.SE, The Swedish Association of Unix Users
Börje Josefsson, EurOpen.SE
Lars Tunkarns, EurOpen.SE
Lasse Sundström, FUUG, The Finnish Unix User Group
Vidar Bakke, NUUG, The Norwegian Unix User Group
Kristen Nielsen, The Danish Unix User Group, DKUUG
Martin Wahlén, SSLUG, Skåne/Sjælland Linux User Group

Please send your abstracts to:
Congrex Sweden AB
Attn: NordU2001
P.O. Box 5619, SE- 114 86 Stockholm
SWEDEN
Phone: + 46 8 459 66 00 Fax: + 46 8 661 91 25
E-mail: congrex@congrex.se

.SE EurOpen USENIX DKUUG EurOpen SSLUG NUUG FUUG

CONNECT WITH USENIX & SAGE

MEMBERSHIP AND PUBLICATIONS

USENIX Association
2560 Ninth Street, Suite 215
Berkeley, CA 94710
Phone: 510 528 8649
FAX: 510 548 5738
Email: <office@usenix.org>
<login@usenix.org>

WEB SITES

<http://www.usenix.org>

<http://www.sage.org>

EMAIL

<login@usenix.org>

CONFERENCES

<conference@usenix.org>
Phone: 949 588 8649

CONTRIBUTIONS SOLICITED

You are encouraged to contribute articles, book reviews, photographs, cartoons, and announcements to *;login:*. Send them via email to <login@usenix.org> or through the postal system to the Association office.

USENIX & SAGE

The Advanced Computing Systems Association &
The System Administrators Guild

;login:

PERIODICALS POSTAGE
PAID
AT BERKELEY, CALIFORNIA
AND ADDITIONAL OFFICES

USENIX Association
2560 Ninth Street, Suite 215
Berkeley, CA 94710

POSTMASTER
Send address changes to *;login:*
2560 Ninth Street, Suite 215
Berkeley, CA 94710